ABC INFORMATIQUE

N° 71

COURS D'INFORMATIQUE PRATIQUE ET FAMILIALE

Comment défile un arrière-plan

Du Légo au logo

Les petits Sharp

Micro : des livres à lire

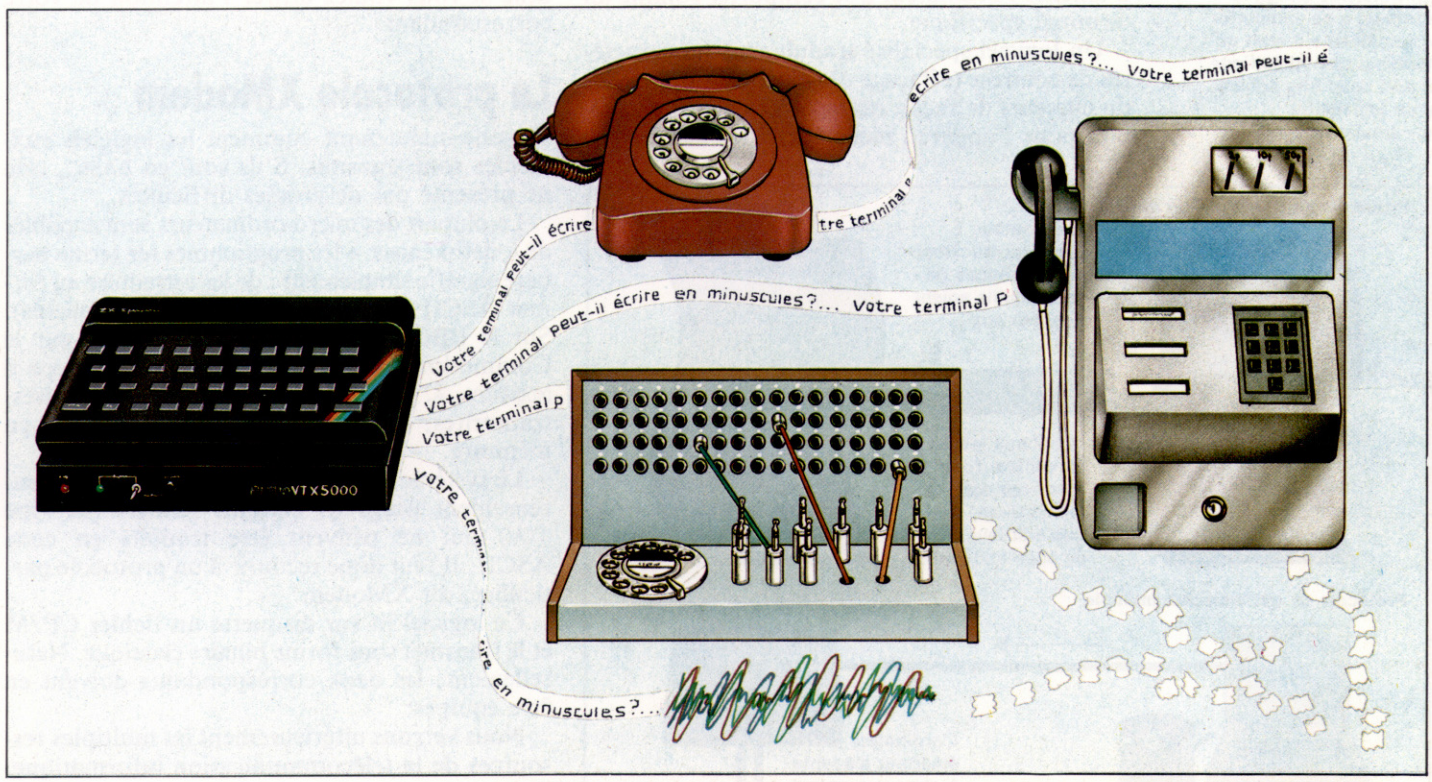
EDITIONS ATLAS

M 6062-71-12,50 F 90 FB - 3,80 FS - $ 1.95

Page manquante
(publicité et colophon)

Comment ça va?

Comment fonctionne un système de communication mettant en œuvre un micro-ordinateur et un modem? Qu'est-ce qu'un « duplex »? C'est ce que nous nous proposons de vous expliquer.



Un système « duplex » a la particularité de permettre la transmission et la réception simultanées des données. Ce n'est pas toujours le cas pour tous les modèles.

Il est possible, de ce point de vue, de faire une comparaison avec le téléphone, où les deux interlocuteurs peuvent parler en même temps, et avec une radio Citizen Band, où l'on ne peut, selon les cas, qu'émettre ou recevoir un message à la fois.

Le principal avantage d'une transmission bidirectionnelle simultanée est qu'il permet de gagner du temps. Si votre correspondant vous transmet un très long menu et que vous avez déjà choisi l'option 1, vous n'aurez qu'à appuyer sur la touche 1, et le système en tiendra compte aussitôt, alors qu'en transmission bidirectionnelle alternative il vous faudrait attendre que le programme soit chargé en mémoire.

La méthode de base est très simple : chaque ordinateur dispose d'une fréquence qui lui est propre. Celui qui appelle a la fréquence « origine », celui qui reçoit a la fréquence « de réponse ». Et, en toute logique, quand un ordinateur transmet des données par l'intermédiaire des lignes téléphoniques, il doit bien sûr savoir quel sera son interlocuteur à l'autre bout du fil. Ce peut être aussi bien un Spectrum qu'un Télétype à 132 colonnes; selon les cas, il disposera, ou non, de la couleur, il acceptera, ou non, les lettres minuscules : car il peut s'agir d'un très gros système ou d'un simple ordinateur domestique.

Protocoles d'accord

On peut traiter le problème de trois façons différentes.

La première est typique de BASICODE, qui ne transmet que les informations accessibles aux terminaux les moins bien équipés; c'est ce qu'on pourrait appeler la méthode du « plus petit commun dénominateur ».

La deuxième consiste à s'adapter aux possibilités du terminal, et à modifier en conséquence la transmission des données.

La troisième consiste à émettre suivant un format à valeur générale; c'est à l'utilisateur de tirer parti des informations reçues, grâce à un logiciel spécialisé.

De ces trois procédés, le premier reste le moins employé : le Télétype est en effet le plus courant

Il n'y a pas d'abonné... On voit parfois, dans les publicités, des ordinateurs portables et des modems acoustiques, alimentés par piles, mis en œuvre dans les conditions les plus difficiles. Mais il est parfois *totalement* impossible de s'en servir : c'est le cas avec les cabines téléphoniques, les standards à commande manuelle (encore en usage dans de nombreux hôtels), ainsi qu'avec des lignes saturées d'électricité statique — ce qui n'est pas rare dans certaines régions un peu à l'écart. (Cl. Steve Cross.)

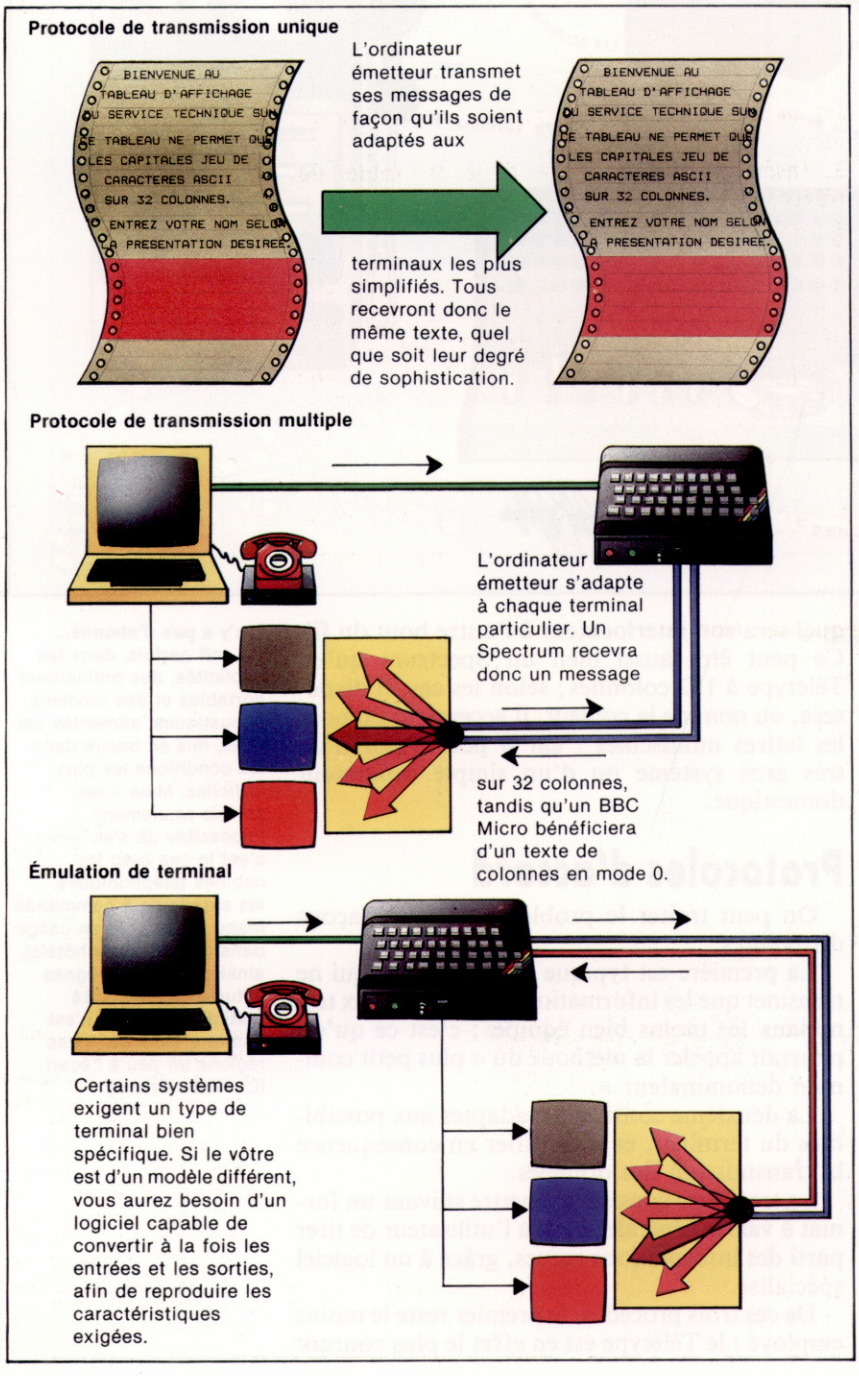
Observer les règles : Il existe trois grandes façons de régler les protocoles de communication. La première est celle du « plus petit commun dénominateur ». La deuxième consiste à modifier les émissions de l'ordinateur émetteur de sorte qu'elles puissent être accessibles à des terminaux très différents. La troisième oblige le terminal à se comporter « comme si » il était un terminal bien spécifique : il doit donc être équipé d'un logiciel dit d'« émulation ».

des terminaux. Or il ne dispose que des majuscules, n'a ni couleur, ni graphisme, ni formatage d'impression, et sa vitesse demeure réduite. La deuxième méthode est la plus commune — c'est par exemple celle de presque tous les réseaux d'utilisateurs.

La troisième est essentiellement réservée aux réseaux commerciaux ou universitaires, et fait usage d'une technique de programmation qu'on appelle « émulation de terminal ».

Comme le nom l'indique, il s'agit simplement d'amener l'ordinateur à se comporter comme un terminal spécifique.

Un logiciel spécialisé traduit tous les caractères de contrôle (effaçage d'écran, positionnement du curseur) de façon qu'ils soient compréhensibles pour l'appareil récepteur. Bien sûr le procédé fonctionne aussi dans l'autre sens. C'est généralement le code de caractères ASCII qui sert de référence. Certains caractères ont une importance toute particulière : ASCII 19 (« XOFF », obtenu par Control-S) interrompt provisoirement la transmission des données, ASCII 17 (« XON », obtenu par Control-Q) la remet en marche, ASCII 10 (« LF », obtenu par Control-J) provoque l'impression d'une ligne sans retour chariot, et ASCII 7 (« BEL », obtenu par Control-G) génère un *bip* sonore, ce qui est très utile si vous voulez attirer l'attention de votre correspondant.



Kevin Jones

Le protocole XModem

Voyons maintenant comment les logiciels eux-mêmes sont transmis. S'ils sont en BASIC, cela ne présente pas de grosses difficultés.

La plupart des micro-ordinateurs sont capables de « détokéniser » les programmes (ce terme barbare signifie simplement : de les retraduire au format ASCII à partir de leur forme compactée). Sur le BBC Micro, il faut faire *SPOOL, sur le Commodore 64, LIST, sur le Tandy, CSAVE, nom du programme, A. Les données sont alors transmises, transcrites, puis de nouveau compactées en mémoire.

Le système d'exploitation CP/M fait malheureusement usage de certains fichiers (de type .COM) qui ne peuvent être traduits en code ASCII. Il faut donc recourir à un protocole particulier, dit XModem.

Ce logiciel lit sur disquette un fichier CP/M et le transmet sous forme binaire classique. Naturellement, les deux correspondants doivent en être équipés.

Nous verrons ultérieurement les multiples ressources de la télécommunication informatique. Bornons-nous pour le moment à signaler les possibilités offertes par un modem.

On appelle *courrier électronique* l'échange de messages entre usagers, par l'intermédiaire d'un gros système qui les garde en mémoire jusqu'à ce qu'ils soient lus par leur destinataire. Le Minitel français en est un bon exemple.

Toutefois, les passionnés d'informatique cherchent surtout à échanger des programmes par téléphone. Un logiciel en BASIC, nous l'avons vu, ne pose aucun problème de transmission : le procédé est simple, rapide et beaucoup plus pratique que des envois par la poste. Vous pourrez même demander conseil à des amis en leur faisant parvenir ce programme génial que vous êtes en train d'écrire, mais qui s'obstine à ne pas tourner !

Un autre usage, encore au stade embryonnaire en France, est celui des réseaux d'usagers. Les possibilités sont très vastes : échange de tuyaux, demande de renseignements, participation à des jeux d'aventures, réception de programmes, etc. Ce phénomène a pris une très grande ampleur aux États-Unis, où de très nombreux particuliers ont monté leur propre réseau, auquel on peut accéder gratuitement, ou presque (mais il faut toujours payer les notes de téléphone !).

Plusieurs moteurs

Nous avons vu comment connecter un servomoteur à un port utilisateur via le système tampon mis au point précédemment. Voici comment commander plusieurs servomoteurs simultanément.

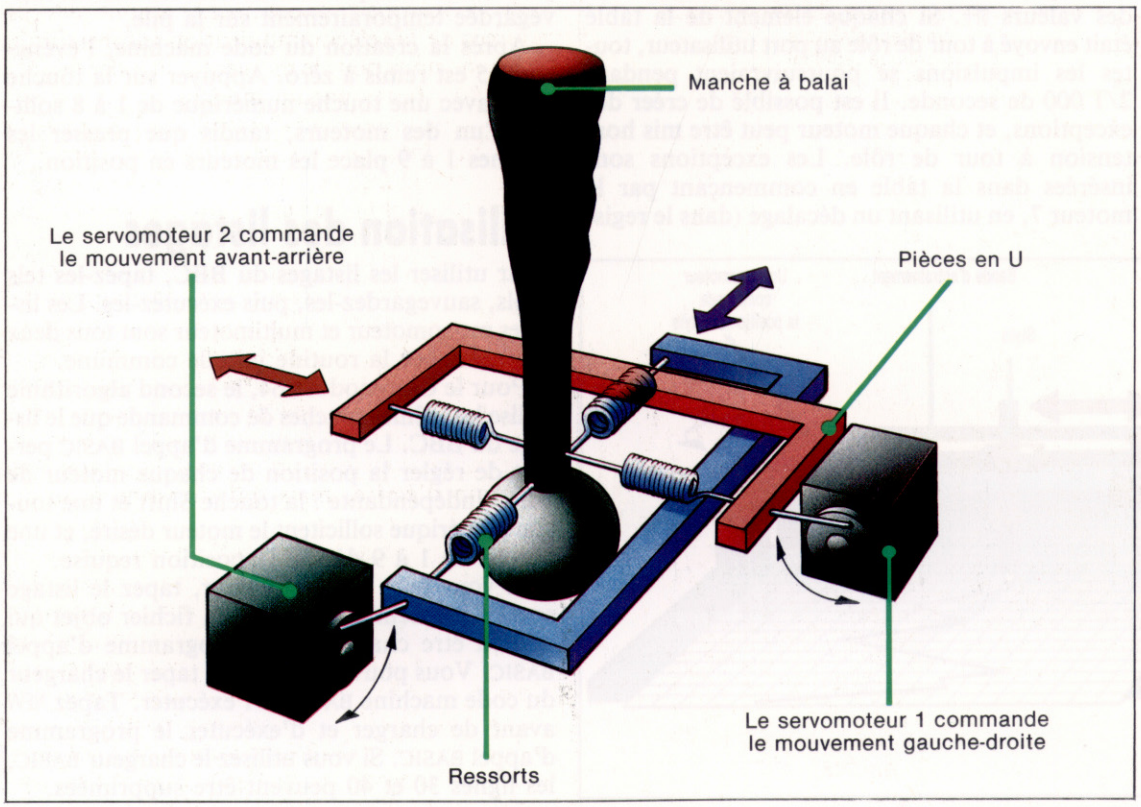
Il y a huit lignes de données qui peuvent être connectées à des moteurs, bien que seules quatre de celles-ci puissent être utilisées par le boîtier tampon que nous avons conçu. Il serait possible d'utiliser les huit lignes en reproduisant les circuits du boîtier tampon pour les quatre autres lignes. Pour commander huit moteurs simultanément, l'algorithme que nous avons mis au point précédemment pour un seul moteur doit être légèrement modifié. Les impulsions commencent toutes au même moment mais la seconde boucle de temporisation est remplacée par une table de référence. 255 adresses de mémoire sont réservées à la table et sont toutes mises à 255 ($FF) initialement.

Les exceptions — lorsqu'un moteur est mis hors tension — sont alors entrées dans cette table. Par exemple, si la ligne de données 2 doit être mise hors tension après un compte de 20, la vingtième entrée de la table passera de la valeur binaire 11111111 ($FF) à la valeur binaire 11111011 ($FB).

Notez que dans le listage d'assemblage cela est fait par l'opérateur AND agissant sur la valeur déjà placée dans la table. Dès que les huit exceptions ont été entrées dans la table, la boucle d'attente est exécutée, mais cette fois chaque élément de la table est associé au port utilisateur au moyen de l'opérateur AND.

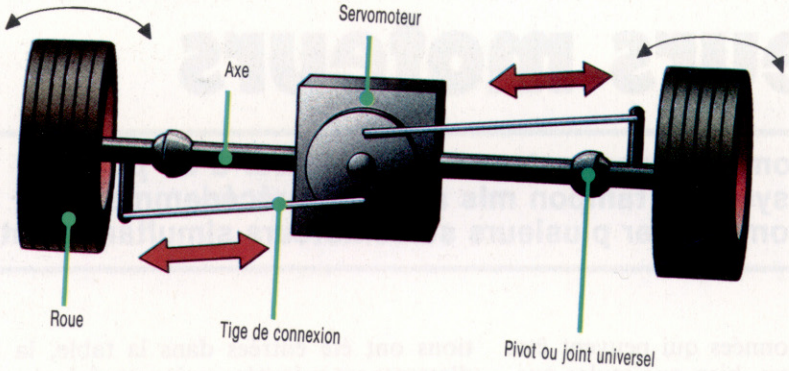
Voici l'algorithme pour la commande de moteurs multiples :

1. Spécifier l'angle de chaque moteur en stockant les angles dans huit octets (de ANGLE + 0 à ANGLE + 7).
2. Mettre tous les bits de données du port utilisateur au niveau élevé pour démarrer simultanément les impulsions.
3. Insérer les exceptions dans la table de référence.
4. Attendre 1/1 000 de seconde.
5. Charger la valeur binaire 11111111 ($FF) dans l'accumulateur. Associer à tour de rôle l'accumulateur avec chaque élément de la table de référence au moyen de l'opérateur AND. Lorsqu'une exception est rencontrée, le bit approprié est mis à zéro. Puisque l'opération AND se poursuit jusqu'à la fin de la table, ce bit demeurera à zéro jusqu'à la fin.
6. Remettre les exceptions de la table à la valeur binaire 11111111, pour la prochaine impulsion.



Réaction du manche à balai
Les manches à balai sont normalement utilisés pour fournir une information directionnelle utilisée par logiciel. Il est possible d'utiliser des servomoteurs pour permettre au logiciel de commande de renvoyer une information au manche à balai. Deux servomoteurs sont utilisés pour pousser et pour tirer le manche dans chaque plan horizontal; ils fournissent ainsi d'importantes données de réaction dans certains simulateurs de vol évolués. Le fait que le manche à balai réagisse dans la main en réponse à un mouvement de commande du pilote améliore la simulation en donnant à l'utilisateur une information tactile en plus des données visuelles. (Cl. Kevin Jones.)

Mécanisme de direction



Dans notre robot d'atelier, la commande directionnelle est assurée par le contrôle individuel de deux moteurs pas à pas bidirectionnels. Le diagramme présente un servomoteur monté directement au-dessus de l'axe : il pousse et tire les tiges connectées aux roues pour diriger le véhicule. Il serait encore possible d'utiliser un mécanisme à support et à pignon, en montant l'engrenage du pignon sur l'axe de rotation du servomoteur.

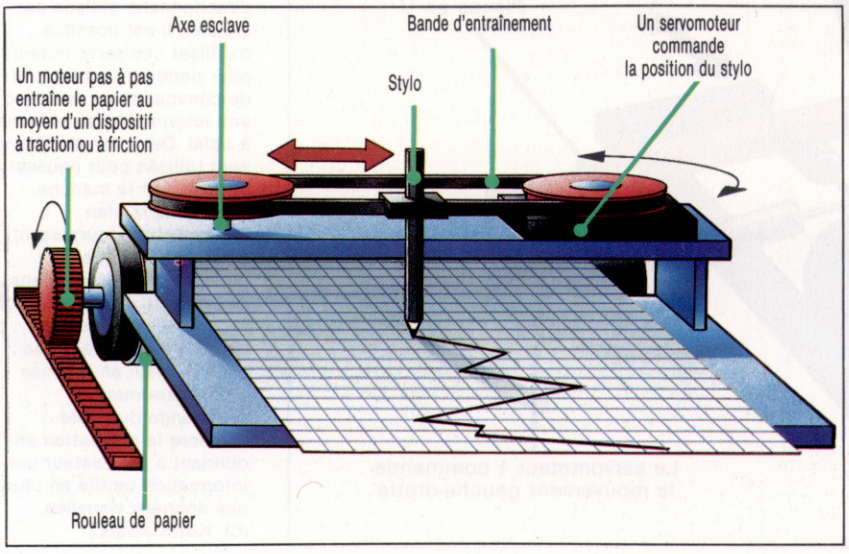
Les listages donnés pour le BBC Micro ont la même routine initiale (lignes 10 à 280) dans les deux programmes. Le premier listage concerne la commande d'un seul servomoteur connecté à l'une des lignes du port utilisateur. Le chronomètreur d'événements est mis en place à l'aide de BASIC. Une procédure d'initialisation assemble la routine de gestion d'événements avant que le programme principal ne l'exécute en autorisant l'événement 5.

Sur le BBC Micro, le système d'exploitation comporte un chronomètre utilisateur au centième de secondes. En le mettant à 2/100 de seconde et en utilisant le vecteur d'événements, le processeur passera au code d'émission d'impulsions au bon moment. Puisque le système d'exploitation fut conçu pour utiliser les événements, le programme n'a qu'à sortir du sous-programme (RTS) et n'utilise pas RTI.

Le second listage, pour la commande de plusieurs servomoteurs, utilise d'abord une boucle BASIC pour initialiser la table de référence avec des valeurs $FF. Si chaque élément de la table était envoyé à tour de rôle au port utilisateur, toutes les impulsions se poursuivraient pendant 2/1 000 de seconde. Il est possible de créer des exceptions, et chaque moteur peut être mis hors tension à tour de rôle. Les exceptions sont insérées dans la table en commençant par le moteur 7, en utilisant un décalage (dans le registre X) proportionnel à la longueur de l'impulsion. Puis la table est envoyée au port utilisateur en adressant chaque élément indirectement, en utilisant cette fois un adressage postindexé (où le processeur ajoute la valeur du registre Y à l'adresse trouvée dans un vecteur de page zéro). Les configurations de bits (exceptions) qui doivent être envoyées au port utilisateur pour commander les moteurs sont produites de la façon suivante. La valeur binaire 01111111 est nécessaire pour mettre le moteur 7 hors tension; 10111111 pour le moteur 6, etc. Ces valeurs sont générées en chargeant $FF dans le registre A et en mettant à zéro le drapeau de report. Puis, tandis que la routine gère chaque exception à tour de rôle, ces bits sont décalés d'une position vers la droite. Lors de la première rotation, le bit de report passe au bit 7, le bit 7 passe au bit 6, et ainsi de suite, jusqu'à ce que le bit 0 remplace le bit de report. La configuration de bits requise pour mettre chaque moteur hors tension est sauvegardée temporairement sur la pile.

Après la création du code machine, l'événement 5 est remis à zéro. Appuyer sur la touche Shift avec une touche numérique de 1 à 8 sollicite l'un des moteurs, tandis que presser les touches 1 à 9 place les moteurs en position.

Traceur
Il est possible de concevoir un traceur en utilisant un servomoteur pour déplacer le stylo, tandis qu'un moteur pas à pas entraîne le papier sous le stylo. Les mouvements angulaires de l'axe de rotation du servomoteur sont traduits en des mouvements linéaires du stylo au moyen d'une bande d'entraînement. Le mouvement du stylo correspond aux changements de la variable de l'axe vertical — par exemple, la température ou la pression barométrique — tandis que le mouvement continu du papier correspond souvent au temps écoulé. (Cl. Kevin Jones.)



Utilisation des listages

Pour utiliser les listages du BBC, tapez-les tels quels, sauvegardez-les, puis exécutez-les. Les listages monomoteur et multimoteur sont tous deux exécutés avec la routine initiale commune.

Pour le Commodore 64, le second algorithme utilise les mêmes touches de commande que le listage du BBC. Le programme d'appel BASIC permet de régler la position de chaque moteur de façon indépendante : la touche Shift et une touche numérique sollicitent le moteur désiré, et une touche de 1 à 9 définit la position requise.

Si vous avez un assembleur, tapez le listage source et assemblez-le dans un fichier objet qui pourra être chargé par le programme d'appel BASIC. Vous pouvez également taper le chargeur du code machine BASIC et l'exécuter. Tapez NEW avant de charger et d'exécuter le programme d'appel BASIC. Si vous utilisez le chargeur BASIC, les lignes 30 et 40 peuvent être supprimées.

Commodore 64
Commande servos multiple

Code source

```
1000 ;++++++++++++++++++++++++++
1010 ;++++++++++++++++++++++++++
1020 ;++                      ++
1030 ;++    COMMANDE CBM      ++
1040 ;++  SERVOS MULTIPLES    ++
1050 ;++                      ++
1060 ;++++++++++++++++++++++++++
1070 ;++++++++++++++++++++++++++
1080 ;
1090 PORT = 56577 ;USER PORT DATA REGISTER
1100 ANGLE=12288  ;ANGLE VALUE LOCATION
1110 ZPAGE=$FB    ;0 PAGE POINTER TO TABLE
1120 ;
1130 *=$0334
1140 ;
1150       SEI        ;INTERRUPTS OFF
1160       LDA $0314 ;EXISTING IRQ VECTOR
1170       LDX $03C4
1180       STA $03C4
1190       STX $0314
1200       LDA $0315
1210       LDX $03C5
1220       STA $03C5
1230       STX $0315
1240 ;
1250 ;++++ INITIALISER TABLE ++++
1260 ;
1270       LDA #$FF
1280       LDY #$00
1290 TABLE
1300       STA (ZPAGE),Y
1310       DEY
1320       BNE TABLE
1330       CLI        ;INTERRUPTS ON
1340       RTS
1350 ;
1360 ;++++ GESTION EVENEMENT ++++
1370 ;
1380       PHP
1390       PHA        ;SAUVE REGISTRES
1400       TYA        ;ON STACK
1410       PHA
1420       TXA
1430       PHA
1440 ;
1450 ;++ DEMARRER IMPULSION, POUR CERTAINS MOTEURS IL EST
1460 ;POSSIBLE DE DEMARRER AVANT DE REMPLIR LA TABLE ET DE
1470 ;REDUIRE LA BOUCLE D'ATTENTE QUI SUIT ++
1480 ;
1490       LDA #$FF
1500       STA PORT
1510 ;++ REMPLIR TABLE D'EXCEPTIONS ++
1520       LDX #$07
1530       LDA #$FF
1540       CLC
1550 EXCEPT
1560       ROR A
1570       PHA        ;BIT PATTERN
1580       LDY ANGLE,X ;GET MOTOR X OFFSET
1590       AND (ZPAGE),Y ;KEEP EXISTING PATTERN
1600       STA (ZPAGE),Y ;BUT MODIFIED FOR MOTOR X
1610       PLA
1620       DEX
1630       BPL EXCEPT
1640 ;++ TABLE CHARGEE ++
1650       LDY #$30
1660 WAIT
1670       DEY           ;FILL IN SOME
1680       BNE WAIT      ;TIME
1690 ;
1700       LDA #$FF      ;ALL PULSES ON
1710       LDY #$00
1720 LOOP
1730       AND (ZPAGE),Y ;BUT MASK OFF WITH EACH
1740       STA PORT      ;TABLE ELEMENT IN TURN
1750       INY
1760       BNE LOOP
1770 ;
1780       LDX #$07
1790       LDA #$FF
1800 CLEAR
1810       LDY ANGLE,X   ;CLEAR ALL EXCEPTIONS
1820       STA (ZPAGE),Y
1830       DEX
1840       BPL CLEAR
1850 ;++ TOUTES LES IMPULSIONS DOIVENT ETRE TERMINEES ++
1860       PLA
1870       TAX
1880       PLA           ;RESTORE REGISTERS
1890       TAY
1900       PLA
1910       PLP
1920       JMP $EA31
```

Programme chargeur basic

```
10 REM **** CHARGEUR BASIC POUR ***
*
20 REM **** SERVOS MULTIPLES ***
*
30 :
40 FOR I=820 TO 922
50 READ A:POKE I,A
60 CC=CC+A
70 NEXT I
80 READ CS:IF CS<>CC THEN PRINT"
CHECKSUM ERROR":STOP
100 DATA120,173,20,3,174,196,3,1
41,196
110 DATA3,142,20,3,173,21,3,174,
197,3
120 DATA141,197,3,142,21,3,169,2,
55,160
130 DATA0,145,251,136,208,251,88
,96,8
140 DATA72,152,72,138,72,169,255
,141,1
150 DATA221,162,7,169,255,24,106,
,72
160 DATA188,0,48,49,251,145,251,
104
170 DATA202,16,243,160,48,136,20
8,253
180 DATA169,255,160,0,49,251,141
,1,221
190 DATA200,208,248,162,7,169,25
5,188
200 DATA0,48,145,251,202,16,248
104
210 DATA170,104,168,104,40,76,49
,234
220 DATA13072:REM*TOTAL DE CONTROLE*
```

Programme d'appel basic

```
10 REM **** SERVOS MULTIPLES ****
20 :
30 ON=0:REM SI CASSETTE ALORS ON=
1
40 IF A=0 THEN A=1:LOAD"MULTISER
V.HEX",DN,1
50 POKE 778,88: POKE 779,3:REM P
OINTER TO EVENT HANDLER
60 POKE 251,0: POKE 252,49:REM S
ET UP ZERO PAGE PTR
70 DDR= 56579:POKE DDR,255:REM A
LL OUTPUT
80 MC=820: SYS MC
90 :
100 GET K$:IF K$="" THEN 100:REM
 AWAIT KEY
110 REM ** MODIFIER POSITION MOTEUR
**
115 AK=ASC(K$)
120 IF AK 48 AND AK<58 THEN POKE
12288+SERVO,VAL(K$)*20
130 IF AK>32 AND AK<48 THEN SERV
O=ASC K$ -35
140 IF K$<>"E" THEN 100:REM 'E'
TO EXIT
```

Commande d'un seul servomoteur

```
  755 REM ******************************
  756 REM   ASSEMBLER LE CODE MACHINE
  757 REM ******************************
10000 DEF PROCinitial
10010 DIM spaceX 200
10020 FOR C=0 TO 2 STEP 2
10030   portb=&FE60
10040   PX=spaceX
10050   angle=PX
10060   PX=PX+1
10070   COPT C
10080   .eventhandler
10090   \save registers first
10100   PHP:PHA:TYA:PHA:TXA:PHA
10110   LDA &&04
10120   LDX &xtimer
10130   LDY &ytimer
10140   JSR &FFF1  \reset timer
10150   LDA &&FF
10160   STA portb
10170   \wait approx.  msec
10180   LDY &&FF
10190   .LOOP DEY
10200         BNE LOOP
10210   \and countout pulse
10220    LDY angle
10230    .LOOP1 DEY
10240         BNE LOOP1
10250   \stop all output pulses
10260   LDA &&0
10270   STA portb
10280   PLA:TAX:PLA:TAY:PLA:PLP
10290   RTS
10300   ]
10310   NEXT C
10320 REM Pointer gestion d'événements
10330 !&220=eventhandler OR (!&220 AND &FFF0000)
10340 ENDPROC
```

Listages BBC Micro

```
10 MODE 0
15 REM **********************************
16 REM  SOLLICITER CE CHRONOMETRE ETC.
17 REM **********************************
20 osbyte=&FFF4
30 AX=&97 :XX=&62 :YX=&FF
40 CALL osbyte:REM Mettre le port B en sortie
50 CLS
60 DIM pX(8)
70 DIM timerX 12,READX 12
80 xtimer=timerX MOD 256
90 ytimer=timerX DIV 256
100 xread=readX MOD 256
110 yread=readX DIV 256
120 PROCinitial
130 FOR IX=angle TO angle+8:angle?IX=128:NEXT
140 t=.02 :REM sec entre impulsions
150 timeX=&FFFFFFFF -(*100) +1
160 timerX?4=&FF    :REM charger octet supérieur
170 !timerX=timeX   :REM solliciter le chrono, autoriser
                     les événements
180 *FX14,5
190 AX=4 :XX=xtimer :YX=ytimer :CALL &FFF1
195 REM **************************
196 REM     CONTROLEUR BASIC
197 REM **************************
200 CLS
210 PRINT"Appuyez sur SHIFT + nombre pour
        sélectionner un moteur"
220 PRINT"Appuyez sur nombre pour choisir l'angle"
225 motor=1
230 REPEAT
240   A=GET
250   IF>A &2F AND A<&3A THEN a=(A-&30)*10+(255/90):angle?
      (motor-1)=a:PRINTTAB(10,motor)!"motor "motor "
      angle "a*(90/255)
260   IF A)&20 AND A &2A THEN motor=A-&20
270   UNTIL 0
280 END
750
```

Commande servos multiples

```
755 REM ************************
756 REM  ASSEMBLE CODE MACHINE
757 REM ************************
760 DEF PROCinitial
770 DIM spaceX 500
780 FOR C=0 TO 3 STEP 3
790   zeropage=&70   :REM libre pour utilisateur
800   portb=&FE60 :osword=&FFF1
810   PX=spaceX
820   angle=PX :PX=PX+8  :REM 8 moteurs potentiels
830   table=PX :PX=PX+256  :REM 256 longueurs d'impulsion
      possible
835   FORIX=table TO table+&100:?IX=&FF:NEXT
840   lowtableX=table MOD 256
850   hightableX=table DIV 256
860   ?zeropage=lowtableX :zeropage?1=hightableX
870   COPT C
880   .eventhandler
890   PHP:PHA:TYA:PHA:TXA:PHA
900   LDA &&04
910   LDX &xtimer
920   LDY &ytimer
930   JSR osword
990   \Démarrer l'impulsion pour certains moteurs il peut
1000   être possible de démarrer avant de réduire ainsi le
       temps d'attente
1010  LDA &&FF :STA portb
1020  \remplir la table d'exceptions
1030  LDX &&7:LDA &&FF :CLC \configuration de bits
1040  .exceptions
1050      ROR A :PHA  \configuration de bits
1060      LDY angle,X  Décalage correspondant à l'angle
          du moteur X
1070      AND (zeropage),Y  \garder la configuration
1080      STA (zeropage),Y  \existante mais modifier pour le
          moteur X
1090      PLA :DEX
1100      BPL exceptions
1110  \table chargée, remplir
1120  LDY &&60
1130  .wait DEY
1140       BNE wait
1150  LDA &&FF  \toutes impulsions ON
1160  LDY &&0
1170  .loop AND (zeropage),Y  \mais masquer chaque élément
1180       STA portb   de la table un élément à la fois
1190       INY
1200       BNE loop
1201  LDX &&7 :LDA &&FF
1203  .clear
1205      LDY angle,X  \mettre à zéro toutes les exceptions
1207      STA (zeropage),Y
1208  DEX
1209  BPL clear
1210  \toutes les impulsions doivent être terminées
1220  PLA:TAX:PLA:TAY:PLA:PLP
1230  RTS
1240  ]
1250  NEXT C
1260  !&220=eventhandler OR (!&220 AND &FFFF0000)
1270  ENDPROC
```

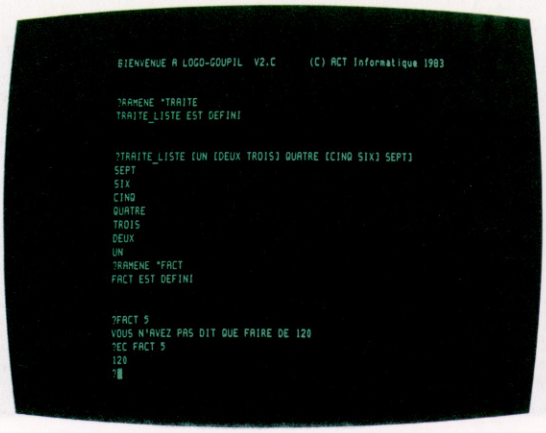
Du Lego au logo

Quand l'ordinateur se fait précieux auxiliaire dans l'éducation des enfants, LOGO est un des langages privilégiés. Cherchons à savoir le pourquoi et le comment.

L'âge de trois, quatre ou cinq ans est celui de la création et de la construction. Entre les blocs à empiler, les Lego à assembler et d'autres cubes à emboîter, les petits doigts acquièrent de l'habileté, tandis que l'esprit se structure. Pour l'aider dans cette tâche qui se poursuivra encore durant des années, l'ordinateur est un instrument merveilleux, à condition de s'en servir correctement. C'est lui qui fournit des « matériaux » à l'« enfant constructeur ».

Si le BASIC est un langage simple pour ceux qui le maîtrisent (et sont en outre un peu familiarisés avec l'anglais), il peut être indiqué pour les adultes qui veulent s'initier à l'informatique (car il s'apprend facilement et permet d'écrire des programmes en peu de temps), il est cependant trop rigide et peu structuré, et ne convient guère aux jeunes enfants. A ceux-ci, il importe de fournir un langage qu'ils puissent apprendre comme leur langue maternelle. L'idée de s'inspirer de la manière naturelle dont un enfant apprend à parler est à la base du LOGO (voir encadré).

LOGO, c'est à la fois une conception de la pédagogie et la famille de langages de programmation allant de pair avec elle. Toutes deux ont été conçues par un groupe de recherche, dirigé par Seymour Papert, au laboratoire d'intelligence artificielle du Massachusetts Institute of Technology (MIT). De même que le nourrisson apprend à parler en progressant dans la compréhension à partir de rien, de même l'enfant peut travailler ou jouer avec l'ordinateur, sans présupposer la connaissance de la lecture ni de l'écriture au départ : en retrouvant des mots dans un certain contexte, il parvient à leur donner un sens, et ces mots lui servent à leur tour pour en comprendre d'autres. Si un langage informatique est appris et couramment parlé dans les premières années de la vie, il sera pratiquement assimilé par l'enfant comme sa langue maternelle, sans réel effort et sans qu'aucun enseignement lui soit imposé délibérément. Cet apprentissage peut alors modifier la façon dont l'enfant abordera le reste des connaissances, en particulier les mathématiques et la logique.



Un écran avenant
Bienvenue ! c'est le premier mot que l'on peut lire sur cet écran du Goupil-LOGO. Une marque de civisme à laquelle les enfants ne sont pas insensibles. (Cl. Goupil.)

Selon Seymour Papert, le rôle de l'ordinateur est celui de porteur de « germes » culturels qui s'enracineront dans un esprit en croissance active. L'important est d'acquérir ces germes assez tôt. C'est là ce qui distinguera ultérieurement les « matheux » des « nuls en maths ». Mais la question n'est pas de produire uniquement des mathématiciens. Le rôle de l'informatique, et notamment des langages de type LOGO, est d'aider à comprendre comment nous pensons et comment nous apprenons à penser. L'ordinateur fournit des « objets avec lesquels penser », selon S. Papert. La « tortue » LOGO est un tel objet. Le rôle de l'enfant consiste alors à dire à la tortue ce qu'elle doit faire et, dans un second temps, à lui apprendre des choses, par exemple des mots nouveaux, des procédures. Ainsi est introduite l'idée de programmation.

Alors qu'en E.A.O (enseignement assisté par ordinateur) c'est généralement l'ordinateur qui « programme » l'enfant en lui faisant faire des exercices dont les réponses sont prédéterminées, en environnement LOGO, la situation est renversée : l'enfant, dès le début, maîtrise la machine. C'est lui qui apprend à l'ordinateur à « penser » et, par là, il parvient à retrouver comment il pense lui-même. En retraçant ainsi la genèse de l'acquisition du savoir (les principes déductifs comme la transitivité, les divers principes de conservation, la logique intuitive, les classifications, etc.), l'enfant se fait épistémologue avant la lettre. Peu à peu, il apprend à exercer un contrôle sur un microcosme riche et complexe.

Programmer en LOGO devient un jeu pour l'enfant. Une séance devant l'ordinateur suffit à l'imprégner de l'« univers-tortue », si bien qu'il n'est pas rare qu'après avoir éteint la machine l'enfant prolonge l'exercice en jouant à être la tortue, et demande à l'adulte qui le guide de lui donner des instructions : « Avance 10 » (l'enfant avance de dix pas), « droite 90 » (il tourne à droite d'un angle droit), « recule 4 » (il recule de quatre pas), et ainsi de suite.

Par la suite, les enfants apprendront que l'ordinateur peut aussi manipuler, outre la tortue, des sons pour faire de la musique, et des mots qui les mèneront progressivement à toutes les subtilités de la programmation.

L'ordinateur sans peur et sans reproches

Si l'enfant n'est pas l'esclave passif de l'ordinateur comme il le serait d'un téléviseur, à l'opposé il n'en est pas non plus le maître tout-puissant. La machine ne fera pas tout ce qui lui est ordonné; il faut toujours respecter certaines règles de langage et de structure; il faut apprendre à lui parler, l'« apprivoiser ».

Très vite, l'enfant se rendra compte que l'ordinateur est un objet qui doit être utilisé d'une façon bien particulière pour donner les résultats qu'on en attend.

Au cours de cette période d'apprentissage — qui d'ailleurs durera aussi longtemps qu'il se servira d'un ordinateur — l'enfant ne connaîtra jamais d'échec. Si en BASIC il y a encore des « erreurs » (syntax error ou autres) à corriger, en LOGO il n'y en a même plus.

Tout au plus l'ordinateur demande-t-il aimablement de préciser un terme, de compléter une instruction. Les messages apparaissant à l'écran seront par exemple : Il manque quelque chose après DROITE ou Que dois-je faire avec 36? ou encore XXX n'aime pas recevoir YYY...

De monstre effrayant qu'il était à ses débuts, l'ordinateur, en devenant « micro », s'est fait compagnon domestique, à la mesure de ses utilisateurs, même et surtout des plus jeunes.

Aujourd'hui, grâce à cet outil fantastique, le pouvoir de choisir des méthodes d'enseignement devrait revenir petit à petit aux particuliers. L'éducation deviendra davantage une affaire privée : les bonnes idées en la matière pourront d'emblée être mises en œuvre sans avoir à passer le barrage rigide du système scolaire national. Nous devrions dès lors assister peu à peu à une renaissance de la pensée pédagogique qui est restée trop longtemps l'apanage d'administrations sclérosées.

Donc, parents et éducateurs, tous à vos ordinateurs!

Jean Raby, éd. ATLAS

Patience et informatique
La patience est aussi une qualité dont savent faire preuve les enfants. Il en faut pour assembler des Lego, il en faut aussi pour faire ses premiers pas au « pays informatique ».

Logo, un nouvel univers pour l'enfant

Le langage informatique LOGO a été créé par Seymour Papert (laboratoire d'intelligence artificielle du MIT), en application des idées du psychologue Jean Piaget sur le développement intellectuel de l'enfant. Selon Piaget, les enfants sont eux-mêmes les bâtisseurs de leurs propres structures intellectuelles : « Comprendre, c'est inventer ou recréer par l'invention ».

L'environnement LOGO fournit, par l'intermédiaire de l'informatique, un « micro-monde » sur lequel l'enfant va pouvoir intervenir en développant des stratégies de pensée pour résoudre les problèmes rencontrés et en découvrant, par ce biais, les lois qui régissent ce « micro-monde ».

Pour le plus jeune enfant, il s'agit d'abord de l'« univers-tortue ». Au moyen de commandes simples — AVANCE, RECULE, DROITE, GAUCHE, suivies de valeurs numériques, et RÉPÈTE, POUR, VIDE-ÉCRAN, etc. —, l'enfant crée des formes à l'écran à l'aide de la « tortue » qui est, en fait, un petit curseur triangulaire se déplaçant dans la direction de son sommet, sur l'écran, suivant les instructions qui lui sont données par l'intermédiaire du clavier.

Les programmes suivants pour tracer un carré, écrits respectivement en BASIC et en LOGO, montrent la puissance, la concision et surtout la « convivialité » de ce dernier langage par rapport au premier.

BASIC

```
10 FOR A = 0 TO 10 : SET (0,A) : NEXT A
20 FOR B = 0 TO 10 : SET (B,A) : NEXT B
30 FOR A = 0 TO 10 : SET (B,10 - A) : NEXT A
40 FOR B = 0 TO 10 : SET (10 - B,A) : NEXT B
```

LOGO

```
RÉPÈTE 4 [DROITE 90 AVANCE 10]
```

Cet « univers-tortue » met l'enfant en présence de toute une série de notions formant le cœur même du calcul différentiel : dans le tracé d'un cercle, par exemple, c'est le « changement d'état » (position et direction) qui importe, et non l'état dans l'absolu, comme c'est le cas pour un repère cartésien; c'est en cela que le LOGO favorise tout à la fois la concrétisation des lois abstraites et une familiarisation précoce et active avec des notions mathématiques ou physiques. Par là même se trouvera encouragée l'acquisition de connaissances scientifiques qui s'enracineront dans l'activité intuitive et personnelle du futur étudiant.

La richesse et la puissance de la programmation en LOGO sont liées à la possibilité de définir un mot à partir d'instructions de base, ou « primitives ». Une telle « procédure » équivaut pour l'enfant à « enseigner » un nouveau mot à l'ordinateur. Voici un exemple de procédure :

```
POUR CARRÉ
  RÉPÈTE 4 [DROITE 90 AVANCE 10]
FIN
```

Dès lors, l'ordinateur « sait » ce que signifie « carré ». Les mots ainsi définis peuvent à leur tour être utilisés dans d'autres procédures. Cette particularité débouche notamment sur la récursivité, dont voici un exemple simple :

```
POUR CERCLE
  AVANCE 1
  DROITE 2
  CERCLE
FIN
```

De toutes les notions enseignées aux enfants, la récursivité est bien la plus marquante et la plus enthousiasmante. C'est un processus qui peut se poursuivre indéfiniment — un peu comme le couvercle de ces boîtes de fromage sur lequel est dessinée une boîte plus petite, sur laquelle est dessinée encore une boîte, et ainsi de suite. L'image devenant de plus en plus minuscule à chaque étape. En LOGO, au contraire, les graphiques deviennent de plus en plus complexes à chaque étape, et le processus peut continuer jusqu'à ce que l'écran soit entièrement couvert. De quoi faire rêver les enfants!

Boucles conditionnelles

S'il fallait trouver un équivalent concret d'une boucle en informatique, on pourrait songer à une noria. Mais dans l'un et l'autre cas, il faut savoir arrêter le mouvement.

Table de multiplication jusqu'à l'infini

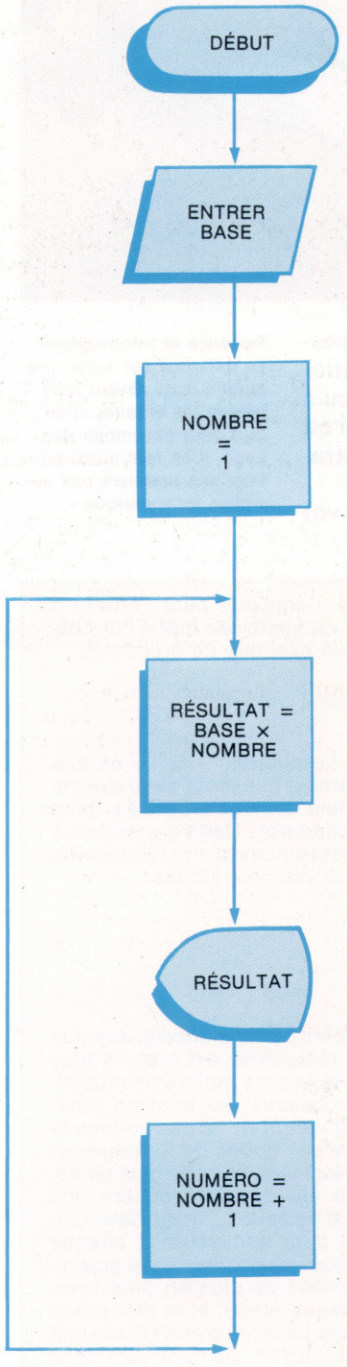


Figure 1

Table de multiplication jusqu'à 10

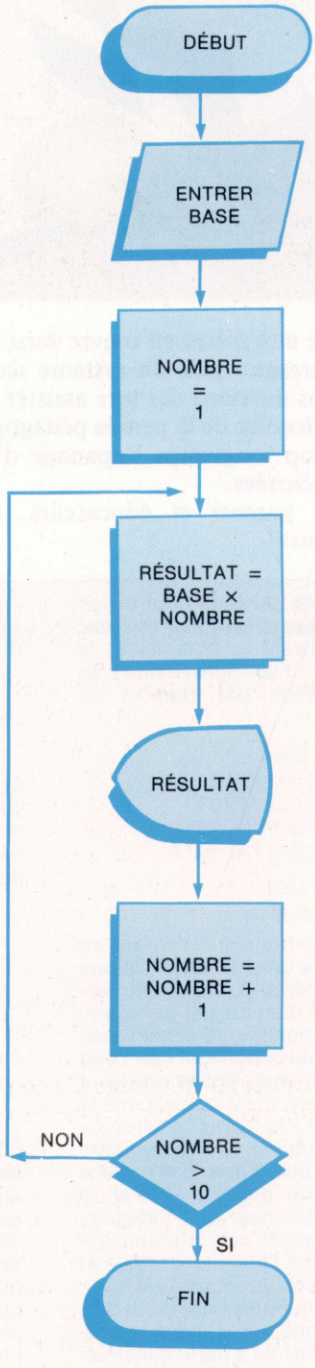


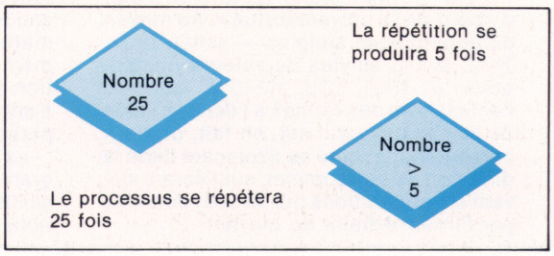
Figure 2

La boucle inconditionnelle — que nous avons déjà vue — a des applications limitées. Supposons que l'on souhaite montrer sur l'écran la table de multiplication d'un nombre quelconque, sans qu'il y ait de limite. L'organigramme prend une forme équivalente à celle apparaissant dans la figure 1. On part, comme base, du nombre de la table de multiplication que l'on souhaite visualiser. La zone NOMBRE correspond alors aux différentes valeurs qui, après une augmentation systématique d'une unité, vont multiplier la BASE. Une suite de résultats apparaît donc visuellement sur l'écran. Il n'y a pas de fin, si ce ne sont les capacités de calcul de l'ordinateur ou encore l'intervention humaine. La personne qui observe le déroulement du programme peut prendre la décision de le stopper.

Les boucles conditionnelles se caractérisent donc de la manière suivante : leurs fins dépendent toujours d'une décision : soit envoyer la séquence de multiplications qui se répètent, soit sortir du cycle. Reprenant l'exemple de la table de multiplication, supposons que l'on veuille visualiser les dix premiers éléments de la table. Le programme correspond à celui indiqué par la figure 2. La nouveauté apparaît dans la présence, pour la première fois, du losange qui est la marque d'une décision. Pour savoir si le nombre, convenablement augmenté après le calcul, puis affiché, a dépassé la valeur 10, nous sommes en présence de deux issues possibles. Si la condition n'est pas remplie, c'est-à-dire si le nombre est encore parmi les dix premiers, le flux retourne au début du cycle; dans le cas contraire, la séquence sera dérivée vers la fin de l'organigramme.

Il est possible, à n'importe quel moment, d'augmenter ou de diminuer l'intervalle, c'est-à-dire le nombre de fois que le cycle va être répété; il suffit simplement de changer la valeur inscrite dans le symbole de décision.

Un même programme peut contenir un nombre indéterminé de boucles, interdépendantes ou non.



Les petits Sharp

Nous avons déjà examiné de près la gamme d'ordinateurs de poche Casio. Jetons maintenant un coup d'œil sur les deux machines Sharp, assez différentes l'une de l'autre.

Sharp produit deux ordinateurs de poche assez différents alors que Casio propose trois machines relativement similaires. Les deux machines Sharp n'ont pas été conçues de façon à créer une concurrence interne : le PC-1251 est le plus petit ordinateur de poche offert sur le marché; le PC-1500A est beaucoup plus gros, plus puissant. Il est l'ordinateur le plus cher offert par ces deux sociétés.

Le Sharp PC-1251 pèse à peine 115 g et mesure 135 × 70 × 10 mm. La taille du clavier n'est que la moitié de celle d'un clavier standard et les touches n'ont que 4 mm de largeur. En plus du clavier alphabétique, on retrouve un clavier numérique qui permet d'utiliser l'ordinateur comme une simple calculatrice.

Le PC-1251 est doté d'un affichage à cristaux liquides de 24 caractères. Celui-ci peut être ajusté en fonction de différentes conditions d'éclairage au moyen d'une petite roue située sur le côté de l'ordinateur. Le commutateur Mode Selector est situé à la droite de l'affichage. Il existe trois modes de fonctionnement : l'un d'eux permet de définir les fonctions de certaines touches (RSV); un second facilite la programmation (PRO); et le dernier permet d'exécuter un programme BASIC ou d'utiliser la machine comme une calculatrice (RUN). Ce commutateur sert également à mettre l'ordinateur hors tension.

La version de BASIC utilisée est excellente pour une machine de cette dimension. Elle offre des commandes absentes sur certains ordinateurs de poche Casio, comme ASC et CHR$, mais comme le BASIC de Casio elle ne comporte pas l'option ELSE pour l'instruction IF...THEN. Comme les petits ordinateurs Casio, le BASIC du PC-1251 utilise les mêmes noms à une lettre pour les chaînes et pour les variables. Cela signifie que si la variable A a été utilisée pour contenir un nombre, la chaîne A$ ne peut être utilisée. De plus, certains tableaux occupent les mêmes zones de mémoire.

Uniquement neuf messages d'erreur sont produits par cette version de BASIC, et il ne s'agit que de simples lettres qui sont d'une piètre assistance lors de la mise au point des programmes. Comme option supplémentaire, la commande PASS permet de protéger des programmes à l'aide d'un mot de passe. Les numéros de lignes du BASIC sont limités de 1 à 999. En mode d'entrée de programme, deux touches de commande de curseur permettent au programmeur de faire défiler les lignes de programme vers le haut ou vers le bas. Deux autres touches de curseur permettent aussi le défilement latéral dans tous les modes.

Puissance miniature
Avec une RAM CMOS et une unité centrale à 8 bits, un clavier QWERTY, le langage BASIC et une gamme de périphériques optionnels, ces « calculatrices » Sharp peuvent revendiquer l'appellation de micro-ordinateurs véritablement portables.

Comme il y a très peu de logiciels offerts pour la machine, la plupart des utilisateurs doivent écrire leurs propres programmes. Le manuel d'utilisation offre deux types d'aide. Il constitue d'abord un bon guide pour le BASIC, facile à comprendre. Deuxièmement, il renferme les listages de neuf courts programmes, qui n'ont pas tous nécessairement des applications mathématiques (trouver des racines carrées, des écarts-types, etc.); un programme d'exercice de dactylographie et un programme de jeu sont également offerts parmi d'autres. De plus, Sharp offre à un prix relativement modeste trois bandes contenant une sélection de programmes.

Alors que les ordinateurs Casio peuvent loger jusqu'à dix programmes en mémoire simultanément, le PC-1251 est limité à un à la fois. Il est toutefois possible d'utiliser un programme composé de plusieurs sous-programmes, séparés par des instructions END. Il est possible d'exécuter un sous-programme en utilisant une instruction GOTO avec le numéro de ligne approprié. Il est souvent utile d'avoir plusieurs programmes en mémoire simultanément, puisque la seule façon de charger des programmes consiste à taper leurs noms lorsqu'ils sont nécessaires. De plus, l'ordi-

nateur n'a que 4 K de mémoire, ce qui représente un espace tout juste suffisant pour loger quelques programmes BASIC modestes. Ce qui signifie que la présence optionnelle de l'imprimante et de l'unité à cassette est presque essentielle si l'on prévoit une utilisation sérieuse.

Le Sharp PC-1500A

L'autre ordinateur portable Sharp, le PC-1500A, est une version améliorée du modèle précédent (le PC-1500) et est destiné à des utilisateurs plus sérieux. Le PC-1500A mesure 195 × 85 × 25 mm et pèse 375 g, ce qui est trois fois plus lourd que le PC-1251.

Le PC-1500A est muni d'un meilleur clavier et d'un écran légèrement plus grand que son petit frère. L'affichage à cristaux liquides est légèrement élargi (26 caractères au lieu de 24) et la fonction pratique d'ajustement d'écran du PC-1251 n'apparaît pas sur la plus grosse machine.

La version de BASIC est toutefois considérablement améliorée. Les variables peuvent avoir des noms à deux lettres, et les mêmes noms peuvent être utilisés pour désigner les variables numériques et de chaînes sans causer de confusion. Le BASIC comporte aussi certaines fonctions sonores et graphiques. Il est possible de concevoir des motifs sur l'écran à cristaux liquides avec une résolution de 7 lignes par 156 colonnes. La commande BEEP de l'ordinateur permet de moduler la hauteur et la durée des notes, qui sont raisonnablement fortes.

L'ordinateur possède un calendrier et une horloge intégrés qui peuvent être adressés à l'aide de la variable TIME. Ceux-ci continuent à fonctionner même lorsque l'ordinateur est mis hors tension; il n'est donc pas nécessaire de les régler lors de chaque mise sous tension. Cette fonction serait une option pratique sur tout ordinateur domestique. Le BASIC du PC-1500A comporte plusieurs autres commandes qui rendent la machine plus performante que de nombreux ordinateurs domestiques. Parmi ces commandes, citons la commande ON ERROR GOTO et les fonctions de trace, TRON et TROFF. Il comporte une généreuse série de trente-neuf messages d'erreur pour l'ordinateur standard; seize autres sont disponibles pour des commandes utilisées uniquement avec des unités complémentaires. Mais comme pour le PC-1251, ces messages d'erreur ne sont que des numéros et pourraient être plus utiles.

La rangée du haut des touches alphabétiques inclut des mots clés BASIC, et il est possible de poser une étiquette au-dessus des touches pour identifier les dix commandes. Il est assez étrange que Sharp n'ait pas choisi d'imprimer ces noms de commande sur les touches — l'étiquette est facilement égarée. Il est possible de programmer jusqu'à dix-huit fonctions sur les six touches situées au-dessus du clavier principal.

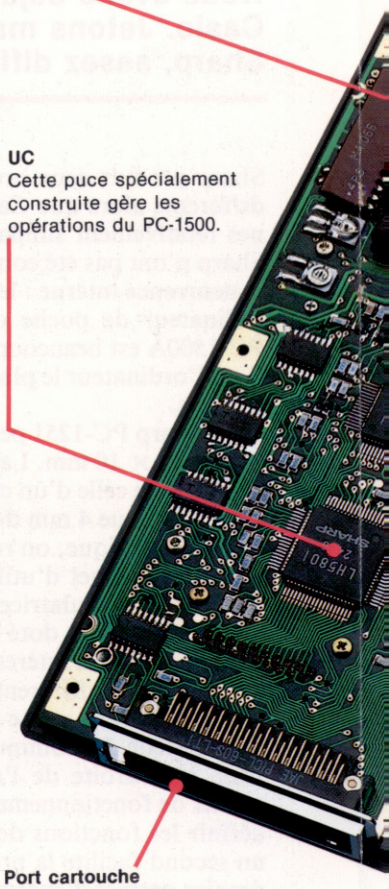
Les utilisateurs du PC-1500A devront écrire la plupart de leurs programmes : peu de sociétés produisent des programmes pour la machine et Sharp ne vend qu'une seule bande de programmes. Un livre accompagnant la machine liste cinquante-trois programmes, avec une variété d'applications dans cinq domaines principaux — mathématiques, statistiques, électricité, travail de bureau et jeux. Ces programmes furent écrits à l'origine pour le PC-1500, mais fonctionnent tous parfaitement sur le PC-1500A (la seule différence entre les deux modèles se situe au niveau de la taille de la mémoire RAM). Le modèle antérieur avait 3,5 K de mémoire, tandis que le PC-1500A en a 8,5 K. Grâce au connecteur de cartouches situé sous la machine, il est possible d'augmenter la quantité de mémoire disponible. Quatre cartouches de mémoire sont offertes mais elles sont toutes assez chères. Les modules de mémoire standard de 4 et 8 K conservent leur contenu uniquement lorsqu'ils sont dans la machine. Deux autres cartouches renferment des piles, leur contenu peut donc être conservé même lorsqu'elles sont retirées de l'ordinateur. Celles-ci ont des capacités de 8 K et de 16 K.

L'unité d'interface d'imprimante-traceur et de cassette est un achat plus judicieux. L'interface de cassette permet de sauvegarder et de charger des programmes au moyen d'une unité à cassette ordinaire — et Sharp offre une unité à cassette compatible. La partie imprimante-traceur de l'unité utilise quatre stylos à bille pour tracer des lettres et des graphiques de qualité en quatre couleurs. Elle est presque identique à de nombreuses autres imprimantes-traceurs offertes sur le marché de l'ordinateur domestique, mais utilise uniquement du papier de 57 mm de largeur, ce qui constitue une restriction importante.

Le BASIC comporte une série complète de commandes permettant d'utiliser l'imprimante-traceur. Celles-ci sont les suivantes : CSIZE pour produire des lettres de diverses dimensions, ROTATE pour imprimer des lettres inclinées ou inversées, COLOR pour sélectionner la couleur du stylo, LF pour déplacer le papier vers le haut ou vers le bas, LPRINT pour imprimer un texte, LCURSOR et GLCURSOR pour déplacer le stylo dans le mode texte et dans le mode graphique, SORGN pour définir l'origine, et LINE et RLINE pour tracer une ligne entre deux points en utilisant respectivement des coordonnées absolues et relatives.

Deux autres dispositifs complémentaires importants sont proposés pour le PC-1500A. L'un est une interface Centronics et RS232, qui permet à la machine de communiquer avec de véritables imprimantes et avec de gros ordinateurs. L'autre périphérique se nomme une « carte logiciel ». Il s'agit d'un grand clavier à touches sensitives qui compte 140 touches qui peuvent être programmées pour exécuter des tâches fréquentes (comme le calcul automatique de totaux dans des applications de tableur).

Bien que le PC-1500A puisse être transformé en un système puissant aux caractéristiques impressionnantes, de nombreux systèmes micro-informatiques sont plus souples et disposent d'une plus large gamme de logiciels. En tentant de se démarquer dans sa catégorie, le PC-1500A risque de ne pas pouvoir concurrencer de nombreuses machines de la catégorie supérieure.



Logement des piles
C'est ici que sont placées les quatre piles nécessaires au fonctionnement de l'ordinateur.

UC
Cette puce spécialement construite gère les opérations du PC-1500.

Port cartouche
Cette interface permet d'installer sur le PC-1500 des modules RAM complémentaires. Elle peut aussi être connectée à l'interface imprimante-traceur ou à une interface RS232C.

Prise d'alimentation
Au lieu d'utiliser des piles, l'ordinateur peut être branché au secteur en utilisant un transformateur adéquat.

ROM basic
Cette puce renferme le BASIC Microsoft utilisé dans l'ordinateur.

Toute une famille de périphériques
Le Sharp PC-1251 s'emboîte dans l'unité imprimante-microcassette CE-125. Celle-ci est composée d'une imprimante thermique (24 caractères par ligne) et d'une unité à microcassette. Les dimensions de l'unité sont 205 × 149 × 23 mm.
Le Sharp PC-1500 est au centre d'une vaste famille d'équipements Sharp, et grâce à l'interface CE-158 RS232C/parallèle, il est possible de le brancher à presque n'importe quel micro ou gros ordinateur. L'interface imprimante couleur-cassette CE-150 est un traceur à quatre couleurs produisant neuf dimensions de caractères et doté d'une option graphique. Les modules de mémoire CE-151, CE-155, CE-159 et CE-161 constituent une gamme de modules RAM CMOS allant de 2 à 16 K, dont certains renferment une ROM programmable. La carte logiciel CE-153 est un clavier à touches sensitives destiné à une entrée formatée.

Sharp PC-1500A

PRIX
★★

DIMENSIONS
195 × 85 × 25 mm.

POIDS
375 g.

MÉMOIRE
RAM 8,5 K.

ÉCRAN
Affichage à cristaux liquides 1 × 26 caractères.

REMARQUES
Calendrier et horloge intégrés; bon BASIC; extension de mémoire possible; large gamme de périphériques.

Sharp PC-1251

PRIX
★★

DIMENSIONS
135 × 70 × 10 mm.

POIDS
115 g.

MÉMOIRE
RAM de 4 K.

ÉCRAN
Affichage à cristaux liquides 1 × 24 caractères.

REMARQUES
Trois modes de fonctionnement; 18 touches pouvant être définies par l'utilisateur; une unité à cassette-imprimante disponible.

Puces RAM
Ces puces fournissent les 8,5 K de mémoire, dont 6,6 K sont à la disposition de l'utilisateur.

Puces d'affichage
Ces quatre puces gèrent l'affichage de l'écran à cristaux liquides.

Carte de circuits imprimés
Pour rendre l'unité plus compacte, la carte a été divisée en deux parties qui sont connectées au moyen d'une paire de câbles-rubans. Notez le faible nombre de composants, plusieurs ont été intégrés sur les puces CMOS.

Puce d'E/S
Gère les périphériques externes, comme l'unité imprimante-cassette.

Sharp PC-1251

Sharp présente cette calculatrice comme un système de gestion élémentaire, probablement plus destiné à l'ingénieur et au scientifique qu'au gestionnaire. Les touches minuscules du clavier QWERTY rendent l'entrée de texte difficile et fatigante.

Le Sharp PC-1500A

Voici un ordinateur de poche extrêmement souple de la taille d'une calculatrice de poche; sa puissance et son équipement optionnel sont ses meilleurs arguments de vente et il peut réellement remplir un rôle important dans le travail de bureau. Il est cependant probable qu'il ne soit perçu que comme un jouet pour cadre ou comme une calculatrice impressionnante.

Chris Stevens

Petits dessins

Nous allons voir comment créer des écrans graphiques représentant l'unité arithmétique et logique (ALU) et le port manche à balai du micro-ordinateur ZX Spectrum.

Il nous faut d'abord faire défiler les lettres A, L et U jusqu'au centre de l'écran, à l'aide de graphismes haute résolution. Sur le BBC Micro, on y parvient en dessinant la lettre en un point de départ donné, puis en l'effaçant ; on la redessine un peu plus loin et on recommence l'opération. La même méthode peut être mise en œuvre pour le Spectrum.

Ce dernier est doté d'une commande DRAW qui n'accepte que des coordonnées relatives — c'est-à-dire définies par rapport au dernier point tracé. Mais c'est précisément ce qu'il nous faut ici.

PLOT nous permettra de dessiner un point de départ, puis nous créerons la lettre entière par une série de DRAW, en changeant simplement les coordonnées du point d'origine. Pour effacer, on redessine la lettre telle qu'elle est, mais en inversant les couleurs, grâce à INVERSE 1 (effet activé), puis INVERSE 0. Pour chaque position occupée par la lettre considérée, celle-ci sera donc tracée deux fois : l'une avec INVERSE 0, pour faire apparaître la forme de la lettre, et l'autre avec INVERSE 1, pour qu'elle disparaisse.

La lettre A vient de la gauche de l'écran. Toutes nos instructions seront placées au sein d'une boucle FOR...NEXT, dont la fonction est d'accroître la valeur de la coordonnée x du point d'origine.

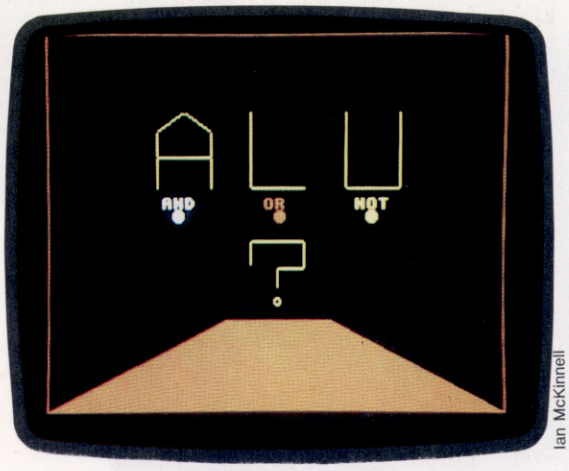
Un premier croquis

A l'intérieur de cette première boucle, nous en imbriquerons une deuxième, dont la tâche consistera à exécuter deux fois les opérations de tracé. X a pour valeur finale 55 : cela correspond à la position finale de A sur l'écran. Cette lettre ne doit plus, alors, être effacée, et nous prendrons soin de ne passer en INVERSE 1 que si x est inférieur à 55. Il va de soi que ces principes des bases vont nous guider également dans le dessin des lettres L et U. Celles-ci viendront respectivement se dérouler sur l'écran, depuis le bas pour la première lettre et depuis la droite pour la seconde (en fait les deuxième et troisième des initiales).

Lorsqu'on entend créer une image à l'écran, il est toujours utile de prendre d'abord une feuille de papier et de faire un premier croquis sommaire, un brouillon en quelque sorte, afin de voir quelles seront les coordonnées de chaque forme mise en œuvre.

De plus, toutes les lettres affichées à l'écran (voir ci-contre) devront elles aussi être positionnées en termes de rangées et de colonnes. Notre illustration montre à quoi ressemble notre dessin une fois achevé.

AND, OR et NOT (que nous franciserons en ET, OU et NON) sont placés à l'écran grâce à la commande PRINT AT r,c, dans laquelle r correspond au nombre de rangées (en partant du haut de l'écran) et c au nombre de colonnes (à partir de la gauche). Les boutons sont tracés à l'aide de CIRCLE x,y,r : x et y sont les coordonnées du centre, et r le rayon.



Ian McKinnell

Une fois que les motifs graphiques ont été tracés, le programme attend que vous pressiez une touche avant d'effacer l'écran et de revenir aux valeurs originales de INK et de PAPER. Il repasse ensuite à la routine principale ALU. INKEY$ permet de tester le clavier — le test est répété jusqu'à ce qu'une touche soit pressée. C'est grâce à :

```
4565 GOSUB 7000 : REM S/P ILLUSTRATION ALU
```

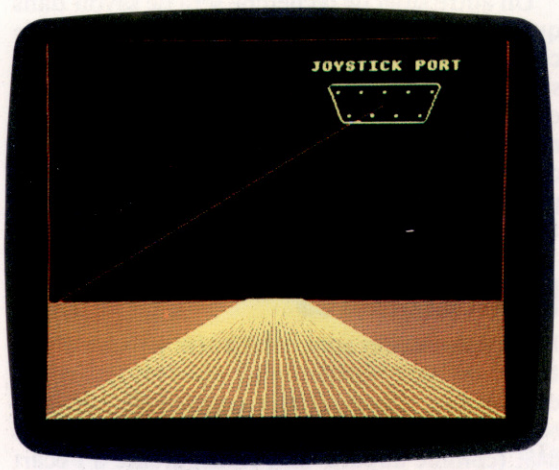
que l'insertion de ce sous-programme dans le listage de Digitaya est assurée.

Le port joystick

Le port joystick crache des rayons laser. Il est représenté graphiquement de façon très simple : les prises sont des points, affichés à l'aide d'instructions PRINT, et le rebord qui les entoure est tracé grâce à des graphismes haute résolution. Une série de lignes de fuite en bas d'écran donne à la scène une certaine profondeur.

Toutes partent d'un point situé sur la ligne d'horizon (déterminé grâce à l'instruction PLOT), et vont jusqu'en bas de l'écran. L'écartement entre elles est d'une unité à l'horizon, et de sept à l'autre extrémité.

La commande DRAW du Spectrum ne prend en compte que les coordonnées relatives, ce qui rend la routine de dessin un peu plus complexe, puisqu'à chaque fois il faut se livrer à de nouveaux calculs. La boucle FOR...NEXT des lignes 8030-8060 s'en chargera : elle intègre les différences d'échelle entre le centre et le bas de l'écran. Là encore, il est bon de dessiner à grands traits notre illustration, de façon à déterminer ses dimensions et ses coordonnées. Le résultat final devrait être comme ceci :



Les rayons laser sont dessinés à partir du port manche à balai : le trait part du centre de ce port et va jusqu'à un point de l'horizon qui, tout comme la valeur de INK, est choisi de façon aléatoire. Il est ensuite effacé en répétant la procédure avec INVERSE 1. Le rayon n'apparaît donc que pour un temps très bref, ce qui donne un effet de clignotement.

On rencontre toutefois un problème. Le trait croise les bords du port manche à balai, et, lorsqu'il est effacé, une partie du port l'est aussi, et doit donc être redessinée.

Le contrôle de la couleur

La ligne d'horizon est également affectée. A cause de la manière dont le Spectrum contrôle la couleur, une partie de cette ligne (celle qui est la plus proche du rayon laser) prendra la même teinte que le trait venu du port joystick : le ZX Spectrum ne peut en effet affecter qu'une seule couleur INK et une seule couleur PAPER à la fois, dans une case d'écran donnée.

Nous serons donc contraints de redessiner non seulement le port manche à balai, mais aussi l'horizon, et ce à chaque effaçage du rayon laser. Le sous-programme employé poursuit les tirs jusqu'à ce que le joueur appuie sur une touche; le programme repasse alors à la routine principale, après avoir redonné à l'écran les valeurs INK et PAPER habituelles.

La ligne suivante doit être insérée dans notre logiciel pour appeler la sous-routine :

```
3845 GOSUB 8000 : REM IMAGE PORT MANCHE A BALAI
```

Nous verrons prochainement comment il est possible d'adapter les deux écrans graphiques précédents au micro-ordinateur Commodore 64.

Écran ALU

```
7000 REM **** S/R IMAGE ALU ****
7010 INK 6: PAPER 0: CLS
7015:
7017 REM **** lettre A ****
7020 FOR x=0 TO 55 STEP 5
7030 INVERSE 0
7040 FOR i=1 TO 2
7050 PLOT x,100
7060 DRAW 0,30
7070 DRAW 15,20
7080 DRAW 15,-20
7090 DRAW 0,-30
7095 DRAW 0,20
7096 DRAW -30,0
7110 IF x<55 THEN  INVERSE 1
7115 NEXT i
7120 NEXT x
7130:
7140 REM **** lettre L ****
7150 FOR y=100 TO 150 STEP 5
7152 INVERSE 0
7155 FOR i=1 TO 2
7160 PLOT 113,y
7170 DRAW 0,-50
7180 DRAW 30,0
7190 IF y<150 THEN  INVERSE 1
7200 NEXT i
7210 NEXT y
7220:
7230 REM **** lettre U ****
7240 FOR x=255 TO 170 STEP -5
7250 INVERSE 0
7260 FOR i=1 TO 2
7270 PLOT x,150
7280 DRAW 0,-50
7290 DRAW 30,0
7300 DRAW 0,50
7310 IF x>170 THEN  INVERSE 1
7320 NEXT i
7330 NEXT x
7340:
7350 REM **** boutons ****
7360 PRINT AT 10,7;"ET"
7370 PRINT AT 10,15;"OU"
7380 PRINT AT 10,22;"NON"
7390 INK 3: CIRCLE 70,80,5
7400 INK 4: CIRCLE 128,80,5
7410 INK 5: CIRCLE 185,80,5
7420:
7430 REM **** point inter ****
7435 INK 6
7440 PLOT 113,45
7450 DRAW 0,15
7460 DRAW 30,0
7470 DRAW 0,-20
7480 DRAW -15,0
7490 DRAW 0,-7
7500 FOR r=6 TO 0 STEP -2
7510 CIRCLE 128,23,r
7520 NEXT r
7530:
7540 IF INKEY$="" THEN GO TO 7540
7550 INK 0: PAPER 7: CLS
7560 RETURN
```

Écran port manette

```
8000 REM **** S/R IMAGE PORT MANETTE ****
8010 INK 6: PAPER 0: CLS
8020 REM **** AVANT-PLAN ****
8030 FOR n=1 TO 31
8040 PLOT 112+n,50
8050 DRAW 7+n-112,-50
8060 NEXT n
8070:
8080 REM **** horizon ****
8085 INK 6: INVERSE 0
8090 PLOT 0,50
8100 DRAW 255,0
8110:
8120 REM **** port ****
8130 PRINT AT 1,18;"PORT MANETTE"
8140 PRINT AT 3,20;". . . . ."
8150 PRINT AT 5,21;". . . ."
8160 PLOT 158,152
8170 DRAW 75,0
8180 DRAW 1,-1
8190 DRAW 1,-1
8200 DRAW 0,-1
8210 DRAW -1,-1
8220 DRAW -10,-25
8230 DRAW -2,-2
8240 DRAW -52,0
8250 DRAW -2,2
8260 DRAW -10,25
8270 DRAW -1,1
8280 DRAW -1,1
8290 DRAW 0,1
8300 DRAW 1,1
8310:
8320 REM **** tir ****
8340 INK RND*7
8350 LET x=RND*255-194
8360 LET y=-86
8365 INVERSE 0
8367 FOR i=1 TO 2
8370 PLOT 194,136
8380 DRAW x,y
8385 INVERSE 1
8387 NEXT i
8390:
8400 REM **** clavier ****
8410 IF INKEY$="" THEN GO TO 8080
8415 INVERSE 0
8420 INK 0: PAPER 7: CLS
8430 RETURN
```

Motifs en treillis

Nous avons déjà étudié divers motifs symétriques. Aujourd'hui, nous abordons les figures à deux dimensions. Voyons comment définir avec LOGO les grilles ou treillis qui sont à la base de ces figures.

Si, au lieu d'effectuer une translation d'une figure selon une ligne, nous effectuons simultanément deux translations non parallèles, nous obtenons un motif en deux dimensions. Nous prendrons le tiret comme unité pour ces motifs.

```
POUR TIRET
   POSEPLUME AVANCE 1 RECULE 1
   LÈVEPLUME
FIN
```

La procédure pour ces translations simultanées se définit de la manière suivante :

```
POUR GRILLE :DÉBUTX :DÉBUTY :UNITÉX
   UNITÉY :ANGLE TRACE CACHETORTUE
   PLUMELÈVE
   DONNEXY :DÉBUTX : DÉBUTY DONNEANGLE 0
   RÉPÈTE 3 [LIGNE :UNITÉX VERSLEBAS
   :UNITÉY :ANGLE]
FIN
```

Cette procédure trace une grille de neuf points sur neuf. Les données en entrée indiquent les coordonnées du point de départ, les unités X et Y, et l'angle pour se positionner sur les points de la ligne suivante. La procédure LIGNE :

```
POUR LIGNE :X
   RÉPÈTE 3 [UNITÉ DONNEX COORX + :X]
   DONNEX COORX - 3* :X
FIN
```

trace une seule ligne de trois unités de long, et repositionne la tortue à son point de départ. Pour l'instant, la procédure UNITÉ est un tiret :

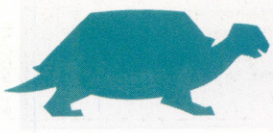
```
POUR UNITÉ
   TIRET
FIN
```

Une autre procédure :

```
POUR VERSLEBAS :Y :A
   DONNEANGLE :A
   AVANCE :Y
   DONNEANGLE 0
FIN
```

positionne la tortue sur la ligne suivante (pour nous, aller « vers le bas »), et redonne l'angle initial. Les cinq types existants de grilles planes sont donnés dans le diagramme avec leurs procédures respectives.

On peut obtenir un très grand nombre de motifs à partir de ces grilles de base, même s'il est probablement plus intéressant de modifier la procédure afin de tracer la ligne de départ selon différents angles plutôt qu'horizontalement.



Un autre sujet de recherche sera de savoir dans quelle mesure on peut améliorer la symétrie de chaque grille en attribuant différents types de symétrie à chaque unité tracée en un point. Il existe dix-sept motifs de cette sorte, et tous figurent au deuxième diagramme. Une méthode évidente pour tracer la commande UNITÉ dans la procédure LIGNE par une procédure qui trace la forme géométrique en ce point.

Formes unités

Les formes unités sont constituées à partir d'une figure de base répétée selon diverses réflexions et rotations. Pour prendre un exemple, définissons une figure de base que nous appellerons BASE (comme auparavant, transparente à l'écart courant et n'utilisant pas de sous-procédures).

```
POUR BASE
   POSEPLUME
   AVANCE 15
   DROITE 90
   AV 5
   RECULE 5
   GAUCHE 90
   RECULE 15
   LÈVEPLUME
FIN
```

Nous pouvons utiliser les procédures développées au dernier numéro pour créer deux nouvelles procédures : FIGURE, et son image en miroir, RÉFLECFIGURE.

```
DÉFINIS «FIGURE TEXTE «BASE
DÉFINIS «RÉFLECFIGURE RÉÉCRIS «BASE
```

Nous pouvons définir maintenant les formes unités. Par exemple, les unités pour le motif 7 seront :

```
POUR UNITÉ 7
   GAUCHE 90 FIGURE DROITE 90
FIN

POUR UNITÉ 17
   DROITE 30
   RÉPÈTE 6 [FIGURE RÉFLECFIGURE DROITE 60]
   GAUCHE 30
FIN
```

Si vous vous reportez à la procédure LIGNE, vous constatez qu'elle exécute la procédure UNITÉ. Aussi, pour tracer le motif 7 par exemple, il faut changer UNITÉ en UNITÉ 7. Nous utilisons pour cela DÉFINIS.UNITÉ 7.

```
POUR DÉFINIS.UNITÉ :NUMÉRO
   DÉFINIS «UNITÉ TEXTE MOT «UNITÉ :NUMÉRO
FIN
```

Exécutons maintenant simultanément les procédures de tracé de grille et de tracé de l'unité. Nous utilisons pour cela une procédure appelée MOTIF :

```
POUR MOTIF :GRILLE :NUMÉRO :PROCÉDURE
   DÉFINIS «FIGURE TEXTE :PROCÉDURE
   DÉFINIS «RÉFLECFIGURE RÉÉCRIS :PROCÉDURE
   DÉFINIS.UNITÉ :NUMÉRO
   EXÉCUTE (LISTE :GRILLE)
   EFFACE FIGURE
   EFFACE RÉFLECFIGURE
   EFFACE UNITÉ
FIN
```

Pour dessiner le motif 17, vous taperiez :

```
MOTIF «HEX 17 «BASE
```

qui tracera une grille hexagonale avec UNITÉ 17 pour chaque point et BASE comme figure de base.

Cette méthode est valable pour tous les motifs à l'exception des motifs 4, 6, 7 et 12. Dans ces derniers cas, la forme unité n'est pas la même en chaque point. Elle subit une transformation (réflexion, rotation, ou les deux à la fois). Une manière de traiter cette différence est d'intégrer ces transformations dans les procédures LIGNE et VERSLEBAS. Aussi nous définirons TRANSFORMATIONX comme la transformation à apporter à la translation horizontale de base, et TRANSFORMATIONY comme celle à apporter à la transformation verticale de base. LIGNE et VERSLEBAS deviennent alors :

```
POUR LIGNE :X
   RÉPÈTE 3 [UNITÉ DONNEX COORX COORX +
   :X TRANSFORMATIONX]
   DONNEX COORX + 3* :X
FIN
POUR VERSLEBAS :Y :A
   DONNEANGLE :A
   AVANCE :Y
   DONNEANGLE 0
   TRANSFORMATIONY
FIN
```

Nous pouvons maintenant définir le motif 7 :

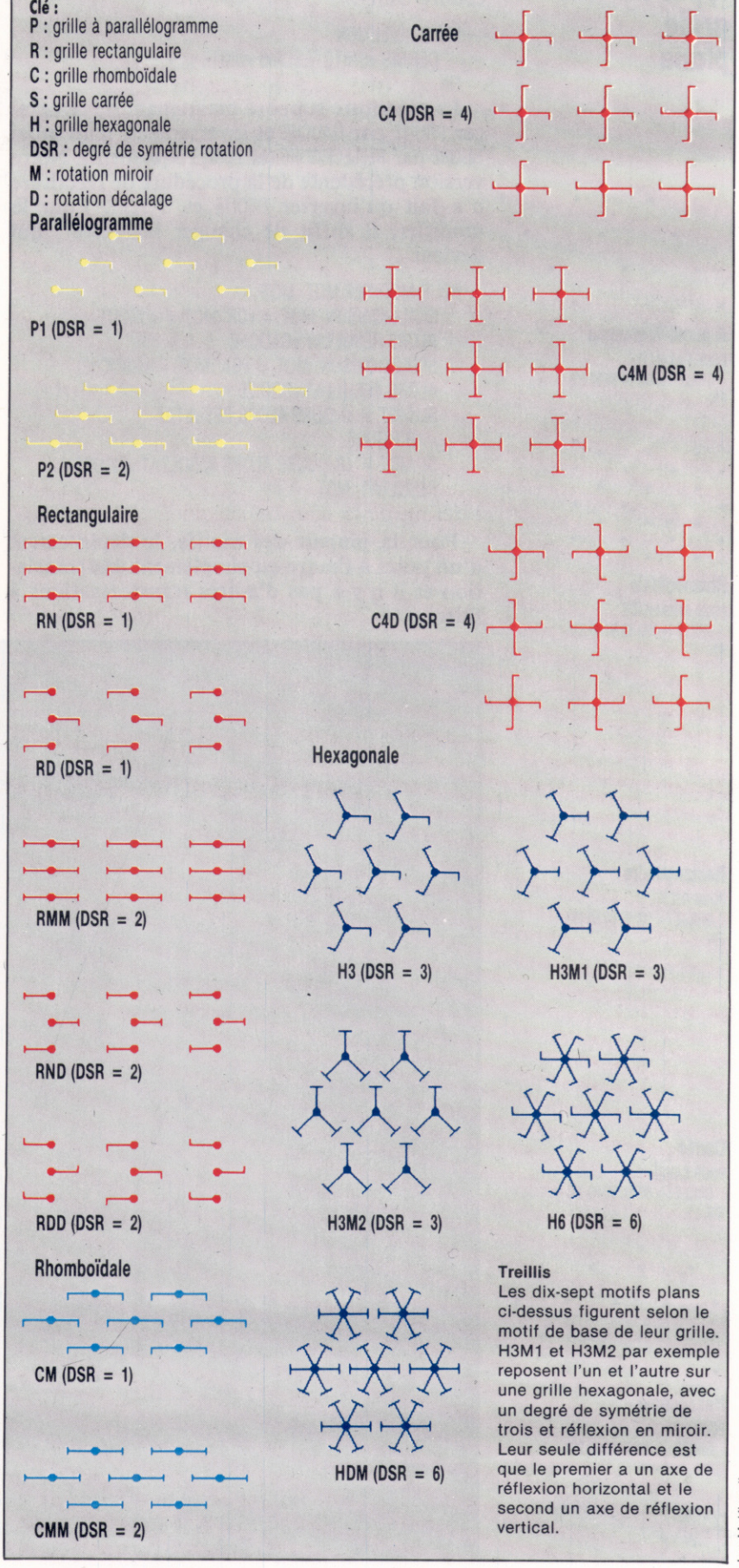
```
POUR MOTIF7 :PROC
   DÉFINIS «TRANSFORMATIONX [ [] [RÉFLEXION DR 180]]
   DÉFINIS «TRANSFORMATIONY [ [] []]
   MOTIF «RÉFLEXION7 : PROCÉDURE
   EFFACE TRANSFORMATIONX
   EFFACE TRANSFORMATIONY
FIN
```

Pour pouvoir utiliser ce dernier, faites MOTIF7 «BÂTONNET. Après avoir exécuté la procédure ci-dessus, TRANSFORMATIONX aurait été définie de la sorte :

```
POUR TRANSFORMATIONX
   RÉFLEXION
   DR 180
FIN
```

Les dix-sept groupes plans

Clé :
P : grille à parallélogramme
R : grille rectangulaire
C : grille rhomboïdale
S : grille carrée
H : grille hexagonale
DSR : degré de symétrie rotation
M : rotation miroir
D : rotation décalage



Treillis
Les dix-sept motifs plans ci-dessus figurent selon le motif de base de leur grille. H3M1 et H3M2 par exemple reposent l'un et l'autre sur une grille hexagonale, avec un degré de symétrie de trois et réflexion en miroir. Leur seule différence est que le premier a un axe de réflexion horizontal et le second un axe de réflexion vertical.

Ian McKinnell

Les cinq types de grille plane

A parallélogramme

```
POUR PARALLÈLE
  GRILLE (– 60) 90 80 50 205
FIN
```

Rhomboïdale

```
POUR RHOMBOÏDE
  GRILLE (– 30) 90 80 80 225
FIN
```

Rectangulaire

```
POUR RECT
  GRILLE (– 80) 90 80 50 180
FIN
```

Carrée

```
POUR CARRÉ
  GRILLE (– 80) 90 80 80 180
FIN
```

Hexagonale

```
POUR HEX
  GRILLE (– 30) 90 80 80 210
FIN
```

RÉFLEXION sert à effectuer une réflexion de la forme unité. Elle se définit en réécrivant la procédure UNITÉ :

```
POUR RÉFLEXION
   DÉFINIS «UNITÉ RÉÉCRIS «UNITÉ
FIN
```

La réécriture suppose maintenant de remplacer DROITE par GAUCHE et inversement, mais aussi FIGURE par RÉFLECFIGURE (et réciproquement). Notre version précédente de la procédure de réécriture n'a fait qu'inverser DROITE et GAUCHE. Pour la modifier, il suffit de changer MODIFIE.MOT, qui devient :

```
POUR MODIFIER.MOT :MOT
   SI(AUMOINSUN :MOT = «DR :MOT = «DROITE)
   ALORS RÉSULTAT «GAUCHE
   SI (AUMOINSUN :MOT = «GA :MOT = «GAUCHE)
   ALORS RÉSULTAT «DROITE
   SI :MOT = «FIGURE ALORS RÉSULTAT
     «R.FIGURE
   SI :MOT = «R.FIGURE ALORS RÉSULTAT «FIGURE
   RÉSULTAT :MOT
FIN
```

Pour la plupart des motifs, le déplacement d'un point à l'autre est simplement une translation et il n'y a pas d'autres transformations à faire.

TRANSFORMATIONX et TRANSFORMATIONY n'ont alors rien à faire. MOTIF17 en est un exemple :

```
POUR MOTIF17 :PROCÉDURE
   DÉFINIS «TRANSFORMATIONX [ [] [] ]
   DÉFINIS «TRANSFORMATIONY [ [] [] ]
   MOTIF «HEX 17 :PROCÉDURE
   EFFACE TRANSFORMATIONX
   EFFACE TRANSFORMATIONY
FIN
```

Après avoir vu les possibilités fondamentales, définissez le reste des motifs.

Variante de logo

Pour toutes les versions LCSI :

Utiliser CS pour DESSINE.
Utiliser OR pour AUMOINSUN.
DONNEPOS suivi d'une liste est mis pour DONNEXY.
SI a une syntaxe différente :

```
SI :MOT = FIGURE [RÉSULTAT «R.FIGURE]
```

TEXTE et DÉFINIS ne figurent pas parmi les primitives Atari LOGO, mais le manuel Atari indique une méthode pour les définir.

Réponses aux exercices

1. Pour faire tourner une forme géométrique autour du point X,Y selon un angle de A degrés :

```
POUR ROTATION :X :Y :A
   LÈVEPLUME
   FAIS «ANGLE ANGLE
   FAIS «ANCIENX COORX
   FAIS «ANCIENY COORY
   FAIS «R CARRÉ (:ANCIENX –
   :X) * (ANCIENX – :X) + (:ANCIENY – :Y)
   LÈVEPLUME
   DONNEXY :X :Y
   DONNEANGLE VERS :ANCIENX :ANCIENY
   DR :A
   AV :R
   DONNEANGLE :ANGLE + :A
   LÈVEPLUME
FIN
```

2. Procédure de réécriture réécrivant également les sous-procédures :

```
FAIS «PROCÉDURES.EFFECTUÉES []

POUR RÉÉCRIS :PROC
   FAIS «PROCÉDURES.EFFECTUÉES FMETS :PROC
   :PROCÉDURES.EFFECTUÉES
   RÉSULTAT RÉÉCRIS.PROC TEXTE :PROC
FIN

POUR RÉÉCRIS.PROC :TEXTE
   SI :TEXTE = [] ALORS RÉSULTAT []
   RÉSULTAT FMETS RÉÉCRIS.LIGNE PREMIER
   :TEXTE
   RÉÉCRIS.PROC SAUFPREMIER :TEXTE
FIN

POUR RÉÉCRIS.LIGNE :LIGNE
   SI :LIGNE = [] ALORS RÉSULTAT [] LISTE
   ? PREMIER :LIGNE ALORS RÉSULTAT FMETS
   RÉÉCRIS.LIGNE PREMIER :LIGNE
   RÉÉCRIS.LIGNE
   SAUFPREMIER :LIGNE
   RÉSULTAT FMETS MODIFIE.MOT PREMIER :LIGNE
   RÉÉCRIS.LIGNE SAUFPREMIER :LIGNE
FIN

POUR MODIFIE.MOT :MOT
   SI (AUMOINSUN :MOT = «DR :MOT = «DROITE)
   ALORS RÉSULTAT «GAUCHE
   I (AUMOINSUN :MOT = «GA :MOT = «GAUCHE)
   ALORS RÉSULTAT «DROITE
   SI PROCÉDURE? :MOT ALORS
   SOUSPROCÉDURE :MOT RÉSULTAT MOT «$ :
   MOT
   RÉSULTAT :MOT
FIN

POUR PROCÉDURE? :NOM
   SI NUMÉRO? :NOM RÉSULTAT «FAUX
   SI LISTE? :NOM RÉSULTAT «FAUX
   TEXTE MOT? :NOM
   SIVRAI SI MOT? TEXTE :NOM RÉSULTAT
   «FAUX SINON SI NON (TEXTE :NOM = [])
   RÉSULTAT «VRAI
   RÉSULTAT «FAUX
FIN

POUR SOUSPROCÉDURE :MOT
   SI APPARTIENT? :MOT
   :PROCÉDURES.EFFECTUÉES ALORS STOP
   DÉFINIS (MOT «$ :MOT) RÉÉCRIS
   :MOT
FIN
```

Marche douce

Beaucoup de jeux de type arcades utilisent un défilement d'arrière-plan pour donner l'impression de mouvement rapide. Voici une routine pour faire défiler l'arrière-plan horizontalement le long de l'écran du C64.

La puce contrôleur vidéo (VIC) du Commodore peut déplacer l'écran du C64 d'une valeur pouvant atteindre 8 pixels dans toutes les directions. Le déplacement horizontal est contrôlé par les 3 bits inférieurs du registre VIC à l'emplacement 53270 ($D016). Si l'on affecte à ces trois bits des valeurs de 7 à 0 successivement, on déplace progressivement l'écran d'un pixel vers la gauche. En BASIC, nous utiliserions l'instruction POKE suivante :

```
POKE 53270, (PEEK(53270) AND 248)+P
```

où P est compris entre 0 et 7.

En combinant cette possibilité avec une routine en langage machine qui déplace toutes les données d'écran d'une cellule vers la gauche et introduit une nouvelle colonne de données sur le bord droit, nous pouvons produire un effet de défilement progressif. Afin que les données puissent entrer et sortir également progressivement de l'écran, la largeur d'écran du C64 doit être réduite à 38 colonnes au lieu de 40. Pour effectuer ce changement de mode, le bit 3 du registre de défilement horizontal doit être mis à zéro. En BASIC, on fait le POKE suivant :

```
POKE 53270,PEEK(53270)AND247
```

On peut remettre l'écran à l'état normal (40 colonnes) en mettant le bit 3 à 1.

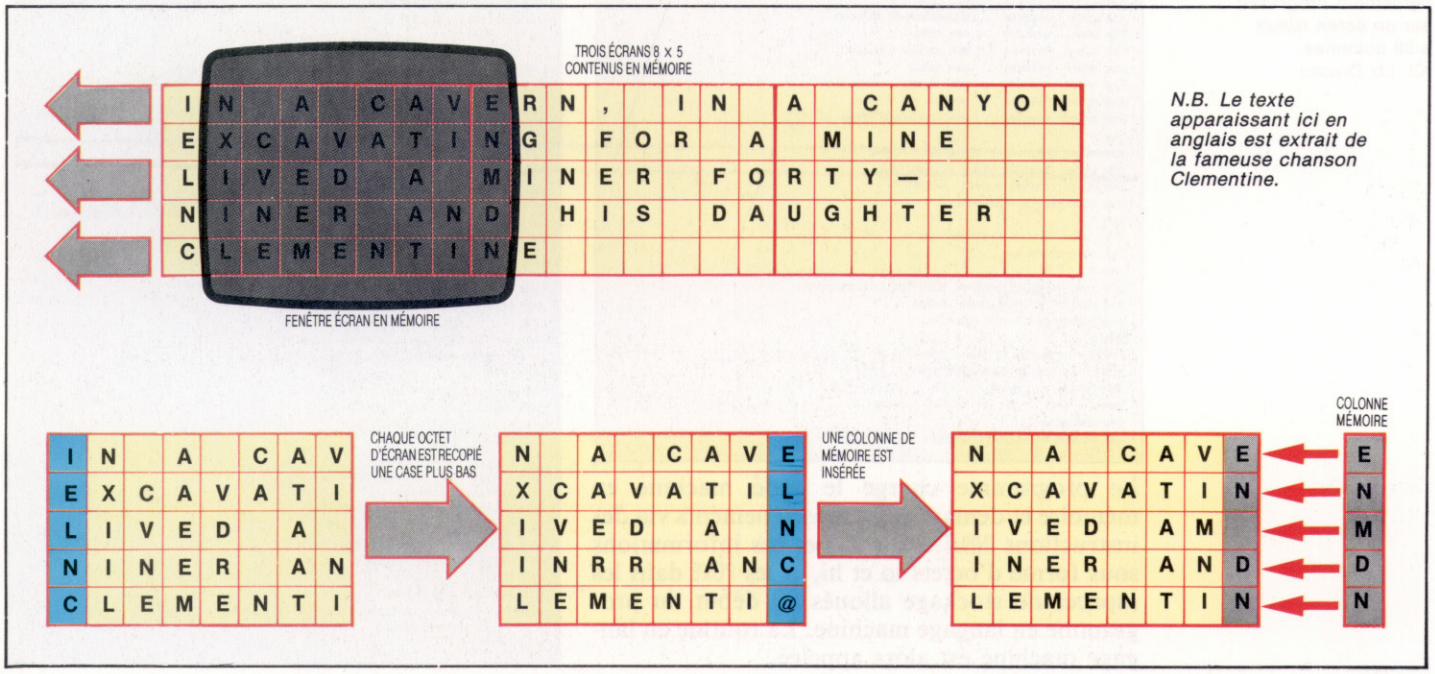
L'organigramme détaille les différentes tâches à accomplir pour produire un défilement horizontal progressif. Il est important de noter que, si nous déplaçons ou insérons des données d'écran, nous devons aussi apporter les changements correspondants aux données couleurs.

Les données d'écran sont normalement contenues dans 1 000 octets consécutifs à partir de l'emplacement 1024 ($0400) : les 40 premiers octets formant la rangée supérieure, les 40 suivants la deuxième rangée, et ainsi de suite. Pour que les données paraissent se déplacer d'une position à gauche, nous n'avons qu'à déplacer chaque octet de donnée dans l'octet en dessous de sa position initiale. Cette partie de la routine emploie les pointeurs de page zéro et l'adressage indirect pour déplacer chaque octet d'écran et donnée de couleur d'un octet plus bas en mémoire.

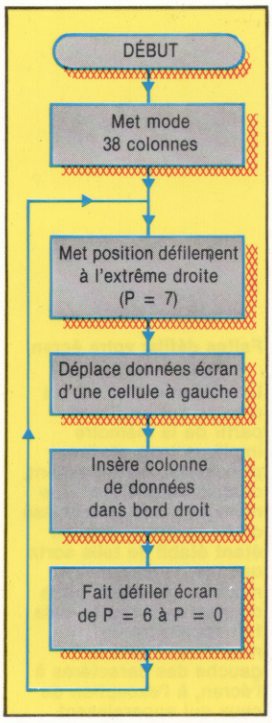
Si nous appelons l'adresse de base de la zone écran SB, alors la dernière cellule de la rangée supérieure sera SB + 39, la dernière cellule de la seconde rangée SB + 79, etc. Pour que les données qui doivent défiler sur l'écran puissent être stockées en mémoire d'une manière analogue à l'écran réel — c'est-à-dire en paquets de 1 000 octets — les données pour l'insertion à la droite de l'écran seront le 41e, 81e, etc., octets de la zone que nous avons réservée pour ces données. Le diagramme suivant explicite cela :

Faites défiler votre écran

Le défilement de motifs d'écran sur le VDU à partir de la mémoire implique trois stades principaux. Premièrement, chaque octet de la zone mémoire écran est abaissé d'une position. L'écran étant établi de telle sorte que les rangées soient mémorisées sous forme d'octets successifs, cela fait apparaître un mouvement vers la gauche des caractères à l'écran, à l'exception de ceux qui apparaissent dans la colonne à l'extrême gauche. Chaque caractère de cette colonne paraît « s'enrouler » vers la colonne de l'extrême droite. Le caractère en haut à gauche de l'écran disparaît pendant ce processus. La seconde phase implique la recopie de la colonne correspondante à partir de la mémoire écran dans la colonne de l'extrême droite. Cela étant, le registre de défilement de la puce VIC peut être manipulé plusieurs fois pour faire défiler un pixel à la fois dans la zone visible. (Cl. Liz Dixon.)



N.B. Le texte apparaissant ici en anglais est extrait de la fameuse chanson Clementine.



Étapes vers un défilement progressif
Pour réaliser un défilement progressif, nous pouvons utiliser une aptitude particulière à la puce VIC du Commodore 64, qui possède des registres spéciaux de défilement permettant de déplacer l'écran visible de sa position normale relativement aux bords. On peut produire des pixels isolés en direction horizontale ou verticale. En combinant cet effet avec la copie de caractères en langage machine, nous pouvons produire un défilement progressif sur un écran réduit à 38 colonnes.
(Cl. Liz Dixon.)

Un pointeur est initialisé pour indiquer l'octet au début de la zone mémoire qui doit défiler sur l'écran. Une fois que la première colonne a défilé, alors le pointeur peut être incrémenté de 1 pour copier une seconde colonne sur le bord droit de l'écran, d'où il peut défiler vers la gauche. Après avoir réitéré le processus 40 fois, un écran complet de données a défilé. Le pointeur doit alors être incrémenté de 960 (1000-40) pour indiquer le début de l'écran suivant.

Ce processus peut être répété pour la zone correspondante de données couleurs. Pour simplifier, nous pouvons faire en sorte que l'adresse de chaque octet de la table couleur ait un décalé constant pour l'adresse de l'octet correspondant dans la table de données écran. Le processus peut être réitéré pour autant d'écrans de données qu'on en a conçu et mis en mémoire.

Afin d'utiliser la routine de défilement, plusieurs éléments d'information doivent passer avant l'appel. La routine doit connaître :

1. L'adresse de début de la zone mémoire contenant les données écran à faire défiler.
2. Le décalé des données couleurs correspondantes.
3. Le nombre d'écrans de données à faire défiler.
4. Une valeur retard, servant à ralentir l'opération de défilement progressif.

Ces données doivent être POKées dans les emplacements réservés dans le programme en langage machine.

Programme basic d'appel

```
10 REM ******************************
20 REM ******************************
30 REM **                          **
40 REM **  PROGR. BASIC D'APPEL    **
50 REM **  DE ROUTINE DEFILEMENT   **
60 REM **                          **
70 REM ******************************
80 REM ******************************
90 :
95 DN=8:REM POUR CASS DN=1
100 IFA=0THENA=1:LOAD"SCROLL.HEX",DN,1
110 POKE55,0:POKE56,32:CLR:REM HAUT DE MEM INF
115 GOSUB 1000:REM ETABLIT AFFICH SIMPLE
120 :
130 LMEM   =49664:     REM DEBUT DE MEMOIRE
140 HMEM   =49665:     REM ZONE
150 LCOFF  =49666:     REM DECALE COULEUR
160 HCOFF  =49667:     REM TABLE
170 NMSCR  =49668:     REM NOMBRE D'ECRANS
190 DELAY  =49669:     REM VALEUR RETARD
200 SCROLL =49670:     REM ADR DEBUT PROGR
210 :
220 PRINT CHR$(147): REM EFFACE ECRAN
230 INPUT "ADRESSE DEBUT DECIMAL";SA
240 HS=INT(SA/256):LS=SA-HS*256
250 POKELMEM,LS:POKEHMEM,HS
260 :
270 INPUT "NOMBRE D'ECRANS";NS
290 POKE NMSCR,NS
300 :
310 INPUT "DECALE DECIMAL POUR TABLE COULEUR";OS
320 HO=INT(OS/256):LO=OS-HO*256
330 POKELCOFF,LO:POKEHCOFF,HO
340 :
350 INPUT"VALEUR RETARD < 256";DV
360 IF DV,255 OR DV<0 THEN 350
370 POKERETARD,DV
380 :
390 SYS SCROLL
400 POKE53270,PEEK(53270)ORB
```

Le programme charge le code machine en mémoire et demande des renseignements via des instructions INPUT. Puis il met ces informations sous forme d'octets lo et hi, et les POKE dans les espaces de stockage alloués au début du programme en langage machine. La routine en langage machine est alors appelée.

Les adresses de début, décalé et nombre d'écrans doivent être spécifiés, bien que les résultats ne soient pas très significatifs si l'on ne met pas de dessins d'écran dans une zone mémoire spécifiée. On peut tester le programme en chargeant et exécutant le court programme BASIC qui établit deux simples écrans de données commençant à l'emplacement 8192. Le décalé à la zone données couleurs est 3 000 octets. Pour faire défiler cette zone données sur l'écran, l'information suivante doit être donnée en réponse aux suggestions du programme d'appel :

1. Adresse de début décimal 8192
2. Décalé couleur 3000
3. Nombre d'écrans 2
4. Retard 255

Chargeur basic

```
10 REM ******************************
15 REM **   CHARGEUR BASIC POUR    **
20 REM **  ROUTINE DE DEFILEMENT   **
30 REM **        HORIZONTAL        **
40 REM ******************************
50 :
60 FOR I=49670 TO 49945
70 READ A:POKEI,A
80 CC=CC+A
90 NEXT
92 READ CS:IF CS<>CC THEN PRINT"CHECKSUM ERROR":STOP
100 DATA173,22,208,41,247,141,22,208
110 DATA174,4,194,160,40,138,72,152,72
120 DATA173,22,208,41,248,24,105,7,141
130 DATA22,208,169,0,133,251,169,4,133
140 DATA252,169,0,133,253,169,216,133
150 DATA254,162,3,160,1,177,251,136
160 DATA145,251,200,177,253,136,145
170 DATA253,200,200,208,241,230,252
180 DATA230,254,177,251,198,252,136
190 DATA145,251,200,177,253,198,254
200 DATA136,145,253,230,252,230,254
210 DATA202,208,213,160,1,177,251,136
220 DATA145,251,200,177,253,136,145
230 DATA253,200,200,192,232,144,239
240 DATA173,0,194,133,253,173,1,194
250 DATA133,254,169,39,133,251,169,4
260 DATA133,252,32,241,194,173,0,194
270 DATA24,109,2,194,133,253,173,1,194
280 DATA109,3,194,133,254,169,39,133
290 DATA251,169,216,133,252,32,241,194
300 DATA162,6,173,22,208,41,248,141,22
310 DATA208,130,24,109,22,208,141,22
320 DATA208,172,5,194,136,208,253,202
330 DATA16,231,173,0,194,24,105,1,141
340 DATA0,194,173,1,194,105,0,141,1
350 DATA194,104,168,104,170,136,240,3
360 DATA76,19,194,173,0,194,24,105,192
370 DATA141,0,194,173,1,194,105,3,141
380 DATA1,194,202,240,3,76,17,194,96
390 DATA162,25,160,0,177,253,145,251
400 DATA202,240,29,165,251,24,105,40
410 DATA133,251,165,252,105,0,133,252
420 DATA165,253,24,105,40,133,253,165
430 DATA254,103,0,133,254,76,245,194
440 DATA96
450 DATA40227:REM=CHECKSUM=
```

Routine établissement d'affichage

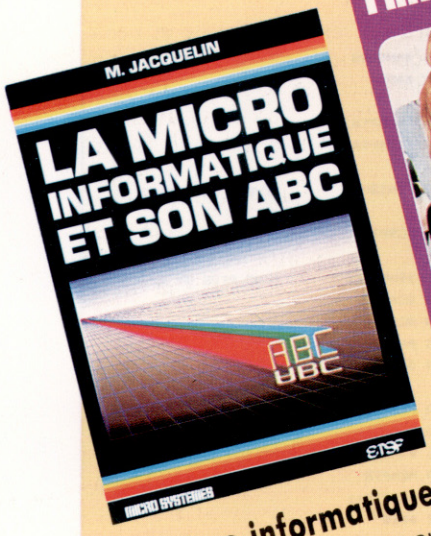
```
1000 REM **** ETABLIT AFFICHAGE ****
1010 CL=3000:REM DECALE TABLE COULEUR
1020 SS=8192:REM DEBUT DE TABLE AFFICHAGE
1030 FORI=SS TO SS+479
1040 POKEI,1:REM CODE ECRAN POUR 'A'
1050 POKEI+CL,1:REM BLANC
1060 POKEI+480,2:REM CODE ECRAN POUR 'B'
1070 POKEI+CL+480,14:REM BLEU CLAIR
1080 NEXT
1085 FORI=SS+960TOSS+999
1090 POKEI,3:REM CODE ECRAN POUR 'C'
1100 POKEI+CL,3:REM BLEU FONCE
1110 NEXT
1999 :
2020 SS=9192:REM DEBUT D'ECRAN SUIVANT
2030 FORI=SS TO SS+479
2040 POKEI,3:REM CODE ECRAN POUR 'C'
2050 POKEI+CL,5:REM VERT
2060 POKEI+480,4:REM CODE ECRAN POUR 'D'
2070 POKEI+CL+480,0:REM NOIR
2080 NEXT
2085 FORI=SS+960TOSS+999
2090 POKEI,5:REM CODE ECRAN POUR 'E'
2100 POKEI+CL,2:REM ROUGE
2110 NEXT
```

Défilement horizontal

```
;++++++++++++++++++++++++++++++
;++++++++++++++++++++++++++++++
;++                          ++
;++  Défilement horizontal  ++
;++       pour CBM 64        ++
;++                          ++
;++++++++++++++++++++++++++++++
;++++++++++++++++++++++++++++++
;
SCRPTR=$FB              ;Ø PAGE
COLPTR=$FD              ;COPY POINTERS
;
MEMPTR=$FD              ;Ø PAGE PTR TO MEMORY
SCRLRG=$D016            ;HORIZ SCROLL REGISTER
SCRNLO=$ØØ              ;SCREEN START LOBYTE
SCRNHI=$Ø4              ;SCREEN START HIBYTE
COLRLO=$ØØ              ;COLOUR START LOBYTE
COLRHI=$D8              ;COLOUR START HIBYTE
BLOCKS=$Ø3              ;3*256 BYTE BLOCKS
EXTRA =$E8              ;EXTRA BYTES TO 1ØØØ
NMCOLS=$28              ;NO OF COLUMNS
NMROWS=$19              ;NO OF ROWS
;
*=$C2ØØ
;
MEMLO   *=*+1           ;START OF MEMORY
MEMHI   *=*+1           ;TO BE SCROLLED
COFFLO  *=*+1           ;OFFSET TO COLOUR MAP
COFFHI  *=*+1
NMSCRN  *=*+1           ;NUMBER OF SCREENS
DELAY   *=*+1           ;DELAY LOOP VALUE
;
;++++ SET 38 COLUMN MODE ++++
;
        LDA SCRLRG
        AND #$F7
        STA SCRLRG
;
;++++ SET 1ST SCROLL POSITION ++++
;
        LDX NMSCRN
NEXSCR
        LDY #NMCOLS
START
        TXA
        PHA             ;PUSH X,Y REGS
        TYA             ;ONTO STACK
        PHA
;
        LDA SCRLRG
        AND #$F8
        CLC
        ADC #$Ø7
        STA SCRLRG
;
;++++ COPY SCRN & COLR ONE LEFT ++++
;
        LDA #SCRNLO
        STA SCRPTR      ;SET UP Ø PAGE
        LDA #SCRNHI
        STA SCRPTR+1    ;POINTERS FOR
        LDA #COLRLO
        STA COLPTR      ;COPY
        LDA #COLRHI
        STA COLPTR+1
;
        LDX #BLOCKS
AGAIN
        LDY #$Ø1
NEXT
        LDA (SCRPTR),Y
        DEY
        STA (SCRPTR),Y
        INY
        LDA (COLPTR),Y
        DEY
        STA (COLPTR),Y
        INY
        INY
        BNE NEXT
;
;++ COPY OVER PAGE BOUNDARY ++
        INC SCRPTR+1    ;INC HIBYTES
        INC COLPTR+1    ;OF Ø PAGE PTRS
        LDA (SCRPTR),Y
        DEC SCRPTR+1
        DEY
        STA (SCRPTR),Y;COPY OVER PAGE
        INY
        LDA (COLPTR),Y
        DEC COLPTR+1
        DEY
        STA (COLPTR),Y
        INC SCRPTR+1    ;INC Ø PAGE PTRS
        INC COLPTR+1    ;AGAIN
        DEX
        BNE AGAIN
;
;++ DO EXTRA BYTES ++
        LDY #$Ø1
ANTHER
        LDA (SCRPTR),Y
        DEY
        STA (SCRPTR),Y
        INY
        LDA (COLPTR),Y
        DEY
        STA (COLPTR),Y
        INY
        INY
        CPY #EXTRA
        BCC ANTHER      ;IF Y<EXTRA REPEAT
;
;++++ INSERT RIGHT COLUMN OF SCREEN ++++
;
        LDA MEMLO
        STA MEMPTR
        LDA MEMHI       ;SET UP Ø PAGE
        STA MEMPTR+1    ;PTRS TO MEMORY
        LDA #NMCOLS-1
        STA SCRPTR      ;SET UP Ø PAGE
        LDA #SCRNHI     ;PTRS TO SCREEN
        STA SCRPTR+1
        JSR COPY4Ø      ;COPY COLUMN
;
;++++ INSERT RIGHT COLUMN OF COLOUR ++++
;
        LDA MEMLO
        CLC
        ADC COFFLO      ;ADD OFFSET TO
        STA MEMPTR      ;COLOUR MAP
        LDA MENHI       ;AND SET Ø PAGE
        ADC COFFHI      ;PTRS
        STA MEMPTR+1
        LDA #NMCOLS-1   ;SET UP Ø PAGE
        STA SCRPTR      ;PTRS TO COLOUR
        LDA #COLRHI     ;RAM
        STA SCRPTR+1
        JSR COPY4Ø      ;DO COPY
;
;++++ SCROLL POSITIONS 6 TO Ø ++++
;
        LDX #$Ø6
MORE1
        LDA SCRLRG
        AND #$F8
        STA SCRLRG
        TXA
        CLC
        ADC SCRLRG
        STA SCRLRG
;
        LDY DELAY       ;COUNT DOWN
MORE2
        DEY             ;DELAY VALUE
        BNE MORE2
        DEX
        BPL MORE1
;
;++++ INCREMENT MEMORY POINTER ++++
;
        LDA MEMLO
        CLC
        ADC #$Ø1
        STA MEMLO
        LDA MEMHI
        ADC #$ØØ
        STA MEMHI
;
;++++ CHECK FOR END OF MEMORY AREA ++++
;
        PLA
        TAY             ;GET X,Y REGS BACK
        PLA
        TAX             ;OFF STACK
        DEY
        BEQ NOJMP
        JMP START
NOJMP
        LDA MEMLO
        CLC
        ADC #$C8        ;ADD 1ØØØ-4Ø
        STA MEMLO       ;TO MEMORY POINTER
        LDA MEMHI
        ADC #$Ø3
        STA MEMHI
        DEX
        BEQ RETRN
        JMP NEXSCR
RETRN
        RTS
;
;++++ COPY EVERY 4ØTH BYTE S/R ++++
;
COPY4Ø
        LDX #NMROWS
        LDY #$ØØ
REPEAT
        LDA (MEMPTR),Y
        STA (SCRPTR),Y
        DEX
        BEQ FINISH      ;EVERY ROW DONE?
;
        LDA SCRPTR
        CLC
        ADC #$NMCOLS    ;INC PTRS BY 4Ø
        STA SCRPTR
        LDA SCRPTR+1
        ADC #$ØØ
        STA SCRPTR+1
;
        LDA MEMPTR
        CLC
        ADC #NMCOLS
        STA MEMPTR
        LDA MEMPTR+1
        ADC #$ØØ
        STA MEMPTR+1
        JMP REPEAT
FINISH
        RTS
```

Livres pour débuter

Voici une sélection de livres pour vous initier à ces étranges et merveilleuses machines que sont les micro-ordinateurs. Ils vous feront entrer dans le cercle des amateurs éclairés, puis des passionnés de micro-informatique.



La micro-informatique et son abc

Tout ce qu'il faut savoir sur l'organisation des micro-ordinateurs en treize chapitres : les systèmes numériques; comptage, addition et soustraction; multiplication et division; les systèmes logiques; additionneur; registre et mémoire; les systèmes programmés; l'unité centrale; instructions et programmes; les systèmes d'entrée/sortie; périphériques; coupleurs; les systèmes d'interruptions et d'accès direct.

Par M. Jacquelin.
256 pages, format 15 × 21 cm.
E.T.S.F.

L'ordinateur et l'informatique en 15 leçons

Tout le monde est concerné par l'informatique, chacun d'entre nous doit donc pouvoir la comprendre et accéder à ses techniques. Cet ouvrage répond à votre attente. Il vous apporte de manière simple les connaissances nécessaires et suffisantes pour dominer cette fameuse informatique et devenir des utilisateurs de l'ordinateur, et non des utilisés.

Par P. Morvan.
224 pages, format 14 × 20 cm.
S.E.C.F., Éditions Radio.

Micro-ordinateurs : comment ça marche?

Voici de manière claire et concise les principes de fonctionnement de tous les éléments qui constituent l'univers des micro-ordinateurs et qu'il vous faut maîtriser.

Par R. Schomberg.
96 pages.
Eyrolles.

Principes des ordinateurs

« Cet ouvrage, destiné à des lecteurs qui ne sont pas tous des techniciens spécialistes, est de lecture aisée : la compréhension est facilitée par des tableaux et des figures, et le désir d'être accessible à tous n'a pas empêché P. de Miribel de fournir, en annexe, à ceux qui le désirent, des précisions scientifiques sur les circuits électroniques, les systèmes de numération et les procédés de calcul numérique... »

Par P. de Miribel.
144 pages.
Dunod informatique.

Page manquante
(publicité)

Page manquante
(publicité)